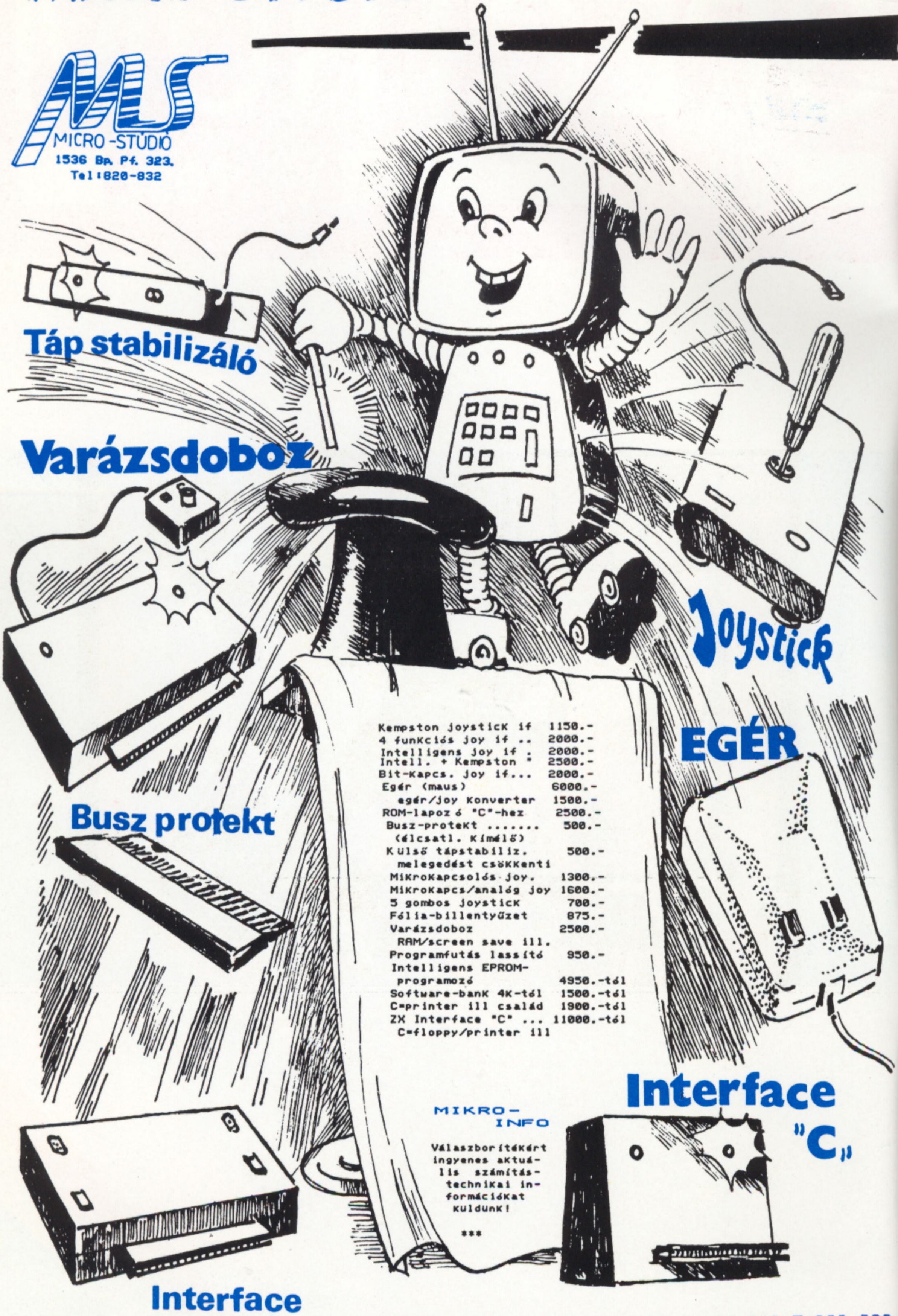
SPECTRUM

VILÁG VI. rész



MICRO-STÚDIÓ



MICRO-STÚDIÓ 1536 Bp.Pf.323 T:820-832

1. SPECTRUMHOZ C= FLOPPY-T !

Régi Spectrumos igény floppy egységet csatlakoztatni a géphez. Sok helyen - főképpen iskolákban - van Commodore rendszer kiépítve. Kézenfekvő a megoldás, ahol lehet, használjunk a Spectrumhoz Commodore floppyt. Hogyan ? Természetesen csak interface-szel. Aktuális kérésnek teszünk eleget, amikor most bemutatjuk a ZX INTERFACE "C" egységet. Mit tartalmaz ez a tekintélyt ébresztő fekete doboz ? Összesen öt interface egységet, amelyek:

- Commodore floppy; - Commodore printer
- Centronics printer; - RS-232 vonal
- és - KEMPSTON joystick

illesztését teszik lehetővé a Spectrumhoz. Érdekessége a készüléknek, hogy a nyomtató illesztő funkción belül összesen öt fajta típust tud kezelni:

- normál EPSON; - színhelyes EPSON
- MPS-801 és ezzel kompatibiliseket
- SEIKOSHA GP-250; - MPS-802 típus

A nyomtató illesztő egyedülálló tulajdonsága, hogy a normál és a dupla nagyságon kívül un. színhelyes copy-t is tud készíteni. Mit is jelent ez a fogalom ? 2:1 nagyságban 1 pont helyett 2x2 azaz 4 pontot kapunk. Ennek a 4 pontnak összesen 5 különböző fedettsége lehet: 4,3,2,1 vagy egy sincs nyomtatva. Ezzel az öt tónussal közelíti az öt különböző mélységű színt. Így lehetővé válik valósághű copy készítése. Nevéhez hűen azt a képet kapjuk a papíron, amit a képernyőn látunk, mivel a rutin nyomtatás közben az attributumokat is figyeli.

Az interface 5 extra szolgáltatást nyújt:

- hideg/meleg RESET

  A hideg-RESET fogalma ismert, míg a meleg RESET "lefagyott", azaz végtelen gépi kódú ciklusba került programot "támaszt fel", visszatérve a BASIC interpreter-hez. Ennek általában az egyik feltétele a rendszerváltozók épsége.

- varázsgomb (NMI)

  Figyelemreméltó a külső hardware megszakítás több célra való felhasználása. Segítségével kiküldhetjük az aktuális képernyő hardcopy-ját ötféle nyomtatóra, 4 vonalra 3 formában. Mód van arra is, hogy a képernyőt floppy-lemezre SAVE-eljük, így egy tetszetős játékprogram pillanatnyi állását elmenthetjük, s azt a későbbiek során egy grafikai programmal egyéni ötletünk szerint továbbszerkeszthetjük.

- színhelyes COPY

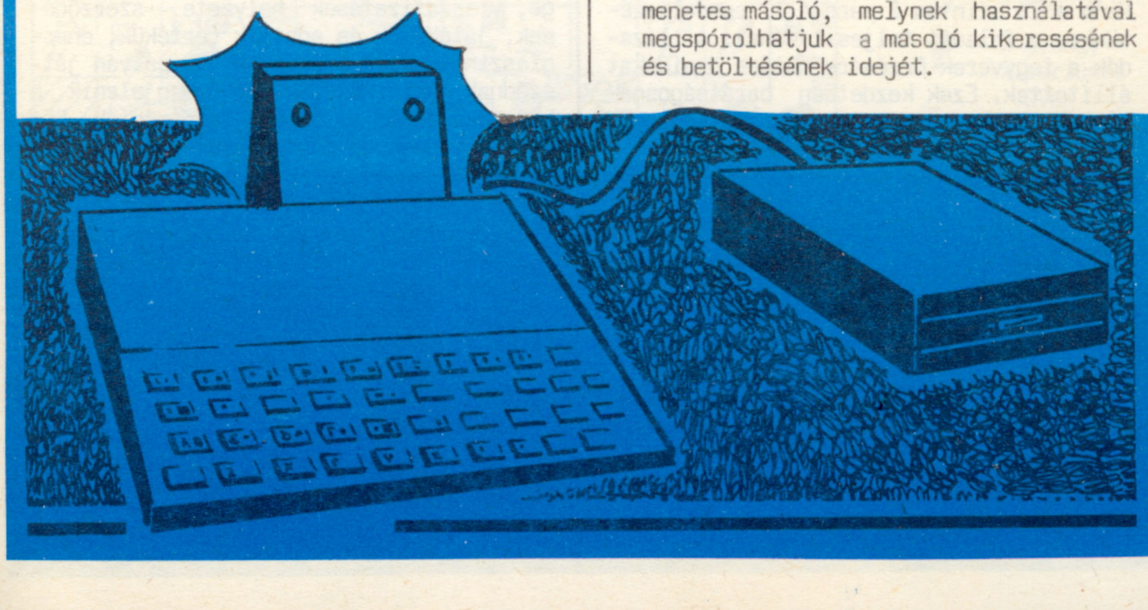
  Alkalmas arra, amint azt a fentiekben leírtuk.

- teljes RAM SAVE

  Szintén a varázsgomb szolgáltatása. Segítségével egy futó (játék)programot archiválhatunk. Visszatöltve ugyanott folytatódik, ahonnan kimentettük. Talán gépi kódú programfejlesztés során a leghálásabb a használata. Fordítás után - futtatás előtt - kimentve a teljes memóriát, programelszállás esetén is vissza tudjuk tölteni a kezdeti állapotot. Ez még akkor is hasznos, ha meleg RESET-tel is visszaállítható lenne a kurzor, mivel nem tudni előre, hogy programunk nem írja-e felül a rendszerváltozókat. Természetesen játékprogramok is kimenthetők ilyen módon vele, esetleg olyan állásban is, amelybe csak elég hosszú idő után lehet eljutni. Ilyen módon egy későbbi időpontban folytathatjuk a küzdelmet.

- Program másolása magnetofonról/ra

  Beépített magnóról-magnóra dolgozó kétmenetes másoló, melynek használatával megspórolhatjuk a másoló kikeresésének és betöltésének idejét.



2. JÁTÉK, POKE, TÉRKÉP

COMBAT SCHOOL
Ocean

Se vége, se hossza az Ocean Imagine Software ház által forgalmazott színvonalas játékoknak. 1987 karácsonyának nagy szenzációja volt ez a játék, grafikája egyszerűen elbűvölő, az ötlet hagyományos. Ebben a játékban az US Marines Corps katonai akadémián kell megfelelnünk a követelményeknek, összesen hét szintet kell teljesítenünk. Először fizikai erőnlétünket teszik próbára, futhatunk keresztül a magas falakon át. A 2. szinten célbalövés következik, majd ismét fizikai próba, futás, ill. úszás. A 4. szinten robottankokra kell tüzelni, nehéz pálya. Az 5. szinten szkander-meccs következik, majd repülő célokra tüzelhetünk. Végül a 7. szinten fegyver nélküli sikeres küzdelemmel végezzük el a tanfolyamot.

ANARCHY
Rack-It

A játék szerzője - Dominic Robinson - több ismert Hewson játék (pl. Uridium, Zynaps) után, most a Rack It színeiben jelent meg az 'ANARCHY' c. játékával. A 'Sentinel 4' nevű bolygót lázadók lepik el, kézenfekvő tehát a feladat, el kell fojtani a lázadást. Ez nem lesz könnyű, de ha elpusztítjuk a fegyvereket, akkor kénytelenek lesznek lemondani a további lázadásról. Először is be kell hatolnunk a komplexumba, majd el kell pusztítanunk a négyzetes alakú fegyver-konténereket. Mind a 16 szinten 2 perc áll rendelkezésünkre a feladat teljesítéséhez. A lázadók a fegyverek őrzésére mutáns droidokat állítottak. Ezek kezdetben barátságosnak tűnnek, de egyre agresszívabbak lesznek. Szép kivitelezés, egyszerű megoldások.

STAR WARS
Domark

"Az erő veled van! Győzd le a birodalom támadó űrhajóit, pusztítsd el a félelmetes halálcsillagot !" A Domark játékötlete a nagysikerű sci-fi film, a 'Csillagok háborúja' alapján készült. Skywalker - a főhős személyében indulunk a küldetésre, az ismert 'X'-szárnyú űrhajóval. Energiapajzsunk a tűzgolyóknak, lézersugaraknak, egyszerű lövedékeknek 9 támadását tudja csak hárítani. Az űrhajó első részében lézerágyúk találhatók, a célzás a kurzor segítségével végezhető el. A támadók raján áttörve akciónkat a halálcsillagon kell folytatnunk. Célunk természetesen ledobni a torpedót a folyosó végén, ennek következtében fel fog robbanni a halálcsillag. Igazán élvezetes játék, hű mása filmbeli 'ősének'.

GRID IRON
Top Ten Hits

Elsősorban az amerikai football-t kedvelőknek ajánljuk ezt a sport-szimulációt, amely próbára teszi üzleti érzékünket is. A játékban egy amerikai football csapat managere-ként le kell játszanunk egy bajnokságot. Bank-egyenlegünk mintegy negyedmillió dollár, ennyivel láthatunk neki tevékenységünknek. A pénzen játékosokat vásárolhatunk, üzletelhetünk. A játék elején beállíthatjuk az edzettségi szintet, és lekérhetünk a képernyőre sok aktuális adatot: egyenleg, kölcsönök összege, visszafizetések helyzete, szerződések, játékosok és adataik (értékük, energiaszintjük). A mérkőzések 3D pályán játszódnak. A mérkőzések végén megjelenik a kiértékelés és az üzleti információk. Célunk a bajnokság megnyerése.

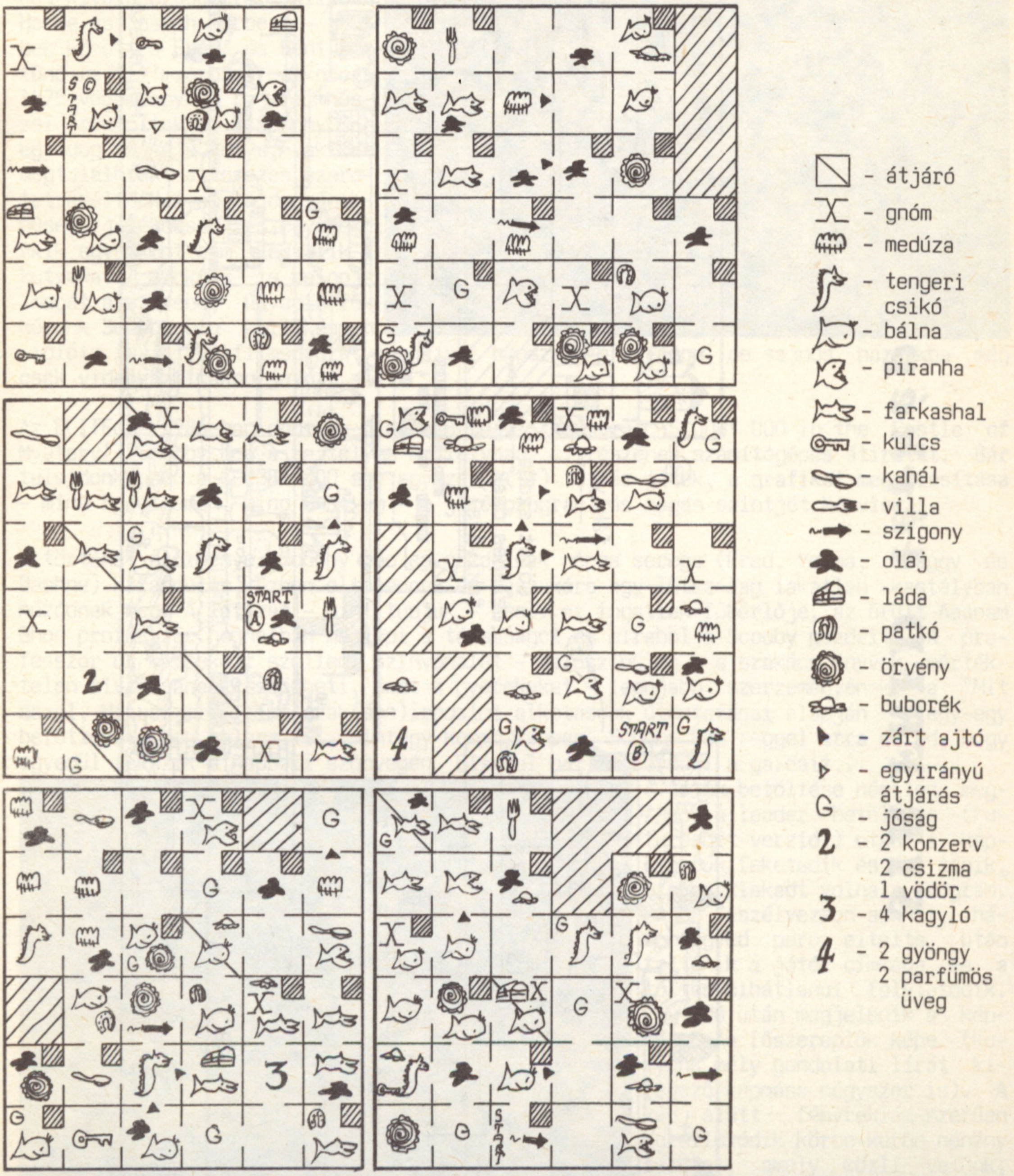


A program két részből áll: egy BASIC betöltőből és egy 48983 byte hosszú fejléc nélküli CODE file-ból. Az örökélethez töltsük be a LOADER-t, majd RESET-eljük a gépet, és írjuk be a következő BASIC programot:

```
10 CLEAR 65399
20 FOR i=65400 TO 65446: READ a: POKE i,a: NEXTi
30 DATA 62,255,221,33,0,64,17,87,191,55,20,8,21,243,62,15,211
40 DATA 254,33,64,5,229,219,254,31,230,32,246,1,79,191,205,107
50 DATA 5,210,0,0,245,62,183,50,58,182,241,195,68,181
60 RANDOMIZE USR 65400
```

Futtassuk a programot (RUN+ENTER) és indítsuk el a magnót.

HYDROFOOL térkép



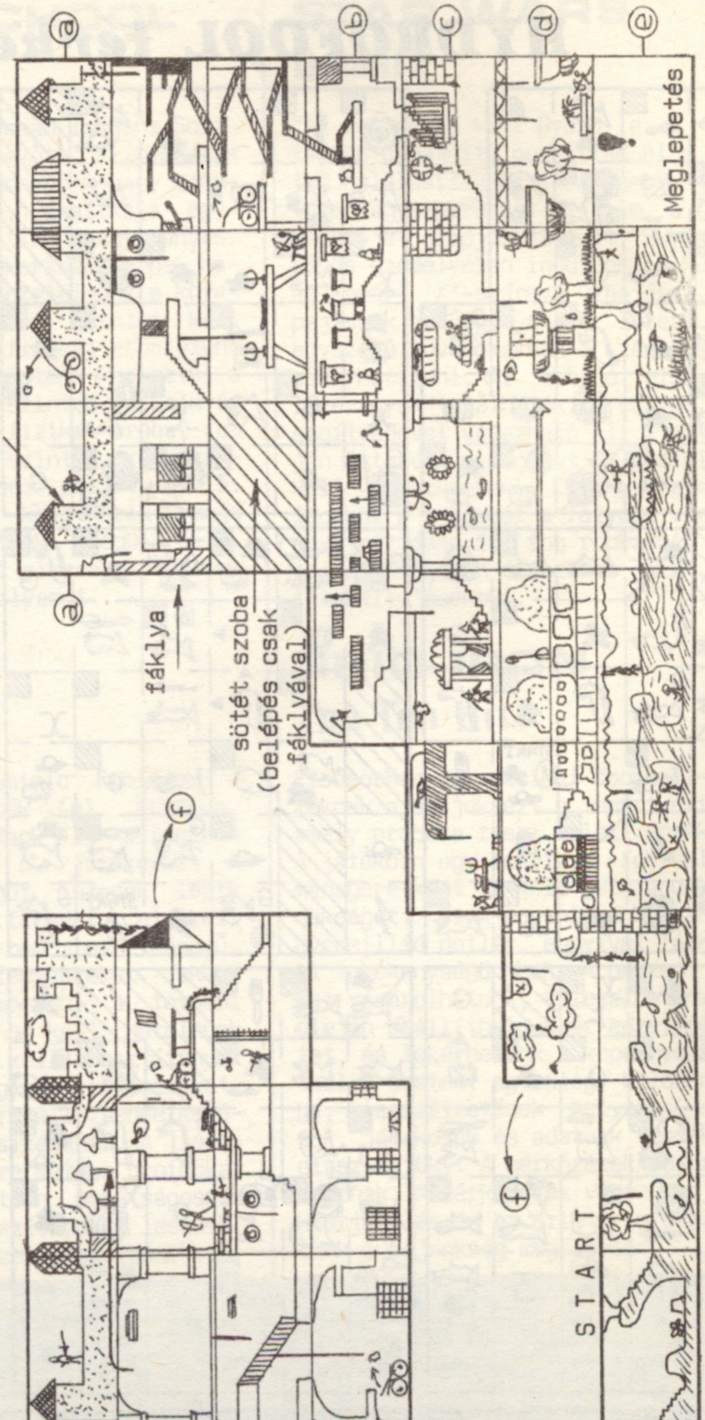
Asterix

A program file-térképe a következő: 404/6912/40960
Az örökélethez töltsük be a LOADER-t és RESET-eljük a gépet, majd írjuk be a következő BASIC programot:

```
10 CLEAR 24575: LOAD"" SREENØ: LOAD"" CODE
20 POKE 36726,0: POKE 36662,0: RANDOMIZE USR 35200
```

Futtassuk a programot (RUN+ENTER) és indítsuk el a magnót.

SIR FRED térkép



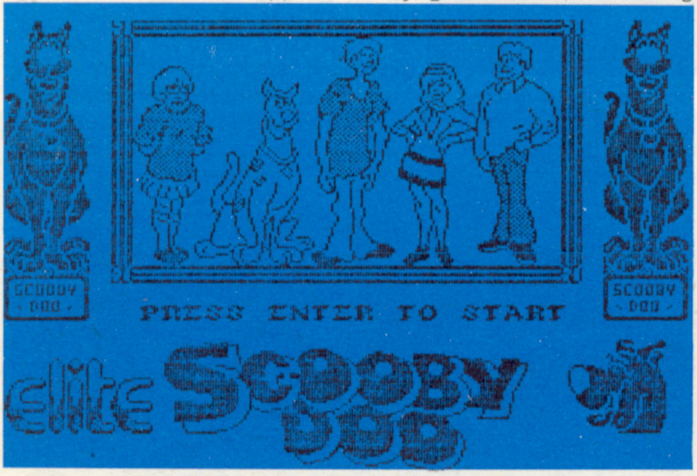
2.1 SCOOBY DOO

A hollywood-i rajzfilmstúdió két kiváló producere, William Hanna és Joseph Barbera - többek között a Frédi és Béni kitűnő rajzfilmsorozat alkotói- 1979 végén egy új rajzfilmhőssel rukkoltak ki. Scooby Doo egy dog fajtájú kutya, akiben megtalálhtók az összes szeretetreméltó kutyatulajdonságok: hűség, játékosság és egyebek. Akik egyébként nem kedvelik a kutyákat, azoknak is belopja magát a szívébe, ugyanis főhősünk Scooby Doo kellőképpen idióta is. A rajzfilmsorozat rendkívül népszerű a világon, de sajnos hazánkba még csak videokazettákon jutott el.



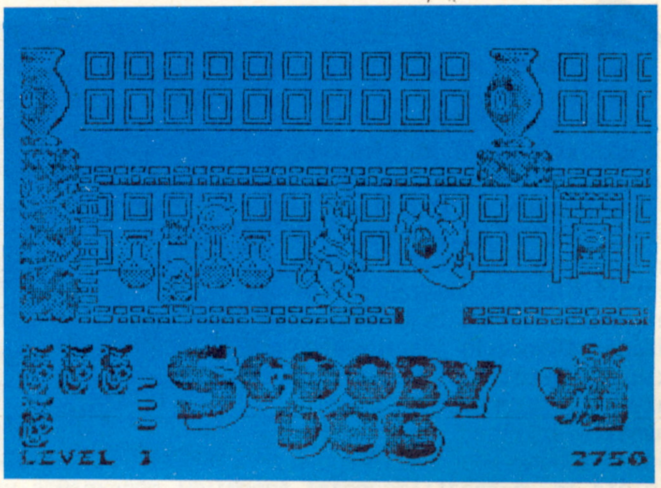
Az ELITEcég 1986-ban dobta a piacra a rajzfilmsorozat 'SCOOBY DOO in the Castle of Mystery' (Scooby Doo a rejtélyes kastélyban) c. részének számítógépes átiratát. Bár tulajdonképpen a SCOOBY DOO egy egyszerű (?!) létrás játék, a grafikai megvalósítása - mind Spectrumon, mind C-64-en - a játékprogramírás magas szintjét képviseli.

A történet előzménye: Scooby Doo és gazdáinak népes serege (Fred, Velma, Shaggy és Daphne) kirándulás közben eltévednek és éjszakára egy látszólag lakatlan kastélyban húzódnak meg. A látszat - mint tudjuk - csal: az ingatlan főbérlője, az őrült Asanam Shoc professzor hajnalban meglepi a társaságot és elrabolja Scooby gazdáit. A professzor úr mérsékelt szellemi színvonalát - valószínűleg - a szakácskönyvek mértéktelen olvasásának köszönheti, mert a gyerekeket - legújabb szerzeményének, a "Mit eszel, Mitesszer?" című szak(ács)irodalmi alkotásnak útmutatásai alapján - egy-egy befőttesüvegben helyezi el, mintegy kompótképpen. Scooby Doo reggel arra ébred, hogy egyedül fekszik a nappali szőnyegén. Elindul hát megkeresni a gazdáit...

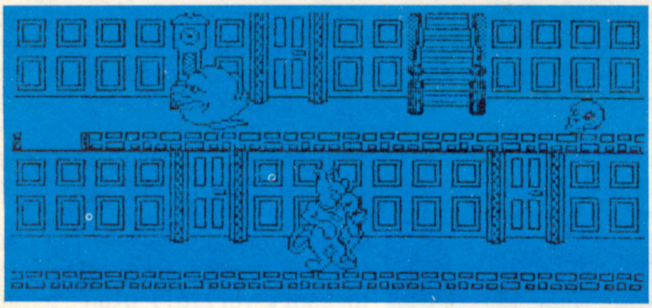


A játék betöltése némileg meglepő: a loader betöltése (Futuresoft-verzió!) után a képernyő kifeketedik és úgy tűnik, mintha kiakadt volna a program. Ez ne feszélyezzen senkit, háromnegyed perc eltelte után feltűnik a játék címképe és a töltés hibátlanul folytatódik. Egy idő után megjelenik a képernyőn a főszereplők képe (kutyánk mély gondolati lírát kifejező képmása négyszer is). A kép alatt fényreklám-szerűen scrollozódik körbe-körbe néhány felirat, amely közli velünk, hogy 'ENTER' megnyomására elindul a játék, 'K' (KEYBOARD) segítségével újra definiálhatjuk az irányító billentyűket, 'J' (JOYSTICK) választásával pedig a joystick-vezérlést állíthatjuk be (Kempston/Interface II/Cursor). Ha valamelyik pályát akarjuk megtanulni, a 'P' (PRACTICE) billentyű megnyomása után az 1-2-3-4 választásával szelektálhatjuk, hogy melyik pályán kívánunk játszani, az '5' billentyűvel visszatérünk a normál játékhoz. Gyakorló üzemmódban nem kerül be a HIGH SCORE-táblázatba az általunk elért eredmény és a sikeres teljesítés után nem kerülünk át a következő pályára.

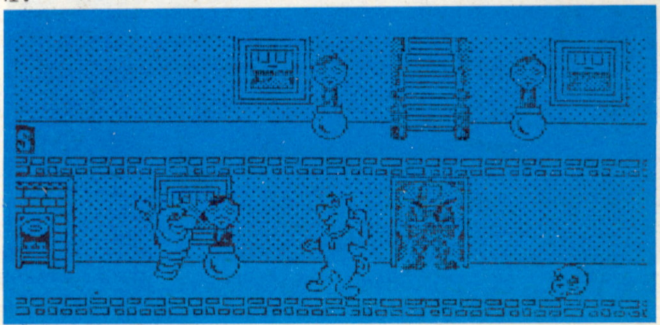
2.1 SCOOBY DOO



Mint már az eddigiekből is kiderült, a játék négy szintből áll. Minden szinten (szinte minden!) egy kompót állapotban leledző gazdinkat kell kiszabadítani, el kell jutnunk az őket tartalmazó kompótosüvegig (a képen is éppen ilyen történés folyik). Ebben a professzor ronda bérencei (szellemes szellemek, kísérteties kísértetek és egyéb egyebek) próbálnak minket megakadályozni. Ezek a folyosókon salfatíroznak le és fel illetve ajtókon, ablakokon és egyéb input/output nyílásokon ugrálnak be. Az ajtók kissé nyikorognak, ha valaki bejön rajtuk, azt egy cuppanó hang jelzi. Erre nem árt odafigyelni, mert a professzor bérenceivel való találkozás egy életünkbe kerül (a képernyő alsó részén látható Scooby-fejekből eltűnik egy). Mint objektív tény, ez elég szomorú dolog, viszont roppant mulatságos látvány: főhősünk egy sikertelen tripla leszúrt Rittberger után, égnek vetett lábakkal eltávozik a Valhallába. Az ellenségek ellen a 'tűz' gomb hathatós védekezést jelent: Scooby egy fenomenális balcsapottal köddé változtatja a támadókat. Minden szinten kétféle rondaság próbál macerálni bennünket, az egyik orrbaverése 50, a másiké 150 pontot jelent. Eredményünket a jobb alsó sarokban lévő Scooby-fej alatt láthatjuk.



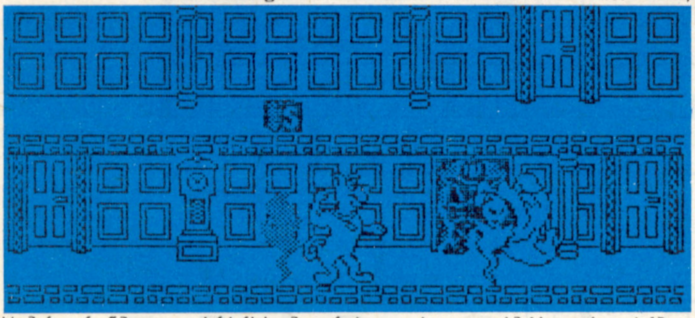
A professzor – úgy látszik – rendkívül rendetlen figura: a – rendhagyó – kompótok elfogyasztása után a magokat szétköpködi az emeleteken. Ebből kifolyólag a folyosókon néhány koponya hever szétszórva, amelyek akadályozzák a továbbjutásunkat. A 'fel' gomb segítségével átugorhatjuk ezeket, de vigyázat, ha ráugrunk a koponyára, spontán módon megpusztulunk. A koponyák átugrálása egyébként is némi körültekintést igényel, mert a derék programozó bácsi úgy intézte a dolgokat, hogy ha futtunkból ugornánk át az akadályt, pont szembetalálkozzunk a professzor csapatának egyik prominens tagjával. Scooby ugrás és a létrákon történő le- illetve felmászkálás közben nem tud ütni, ha ilyenkor szellemmel találkozik nem tud védekezni.



Az első két szinten az ellenségek csak vízszintes irányban (balra/jobbra) mozognak, de a harmadik pályától kezdve már a lépcsőkön és lyukakon keresztül is megpróbálnak a közelünkbe férkőzni. Gyarapodik a konkurrencia populációja is: nem csak kétféle ellenség ijesztget minket, hanem három: néha egy denevér repül be mélyrepülésben a légtérbe. Ezt a repkedő rondaságot sajnos nem lehet lecsapni az egyetlen védekezés ellene az, hogy a 'le' gomb segítségével lehasalunk a földre. Ez a manőver több kínos esetben is hasznos lehet: a bátor Scooby hősiesen eltakarja a szemét, miközben a szőnyeg átható illatú nedvességgel telítődik alatta...

2.1 SCOOBY DOO

A negyedik szint már valóban az élvezetek tárháza lehetne: a légtér telítve repkedő ventilátorokkal, mindenhol a professzor szellemei mászkálnak, néha egy jól megtermett strandlabda üt le a lábunkról, ráadásul a koponyák általában egymásután helyezkednek el; akciónk idővel a gátfutással egybekötött boxmérkőzés nevű jelenséget kezdi idézni és nem tudjuk hirtelen eldönteni, hogy milyen módon haljunk meg...



Mint a fentiekből is látható, Scooby-ra rengeteg környezeti ártalom leselkedik, ami 5 életünket gyorsan elfogyasztja. Életeink számának gyarapítására rendszeresített módszer az "S" betűvel jelölt dobozokban lévő Scooby-snacks (hevenyészett magyar fordításban:Scooby-létyók) elfogyasztása - egy blökivel többen leszünk. Valószínűleg a játékkal párhuzamban zajlik a kastély renoválása, mert a padlón néhol lyukak találhatóak, amelyeken - a gravitációra vonatkozó idevágó passzusok értelmében - érdekes módon lepotyogunk (általában egy ellenség ölébe). Ez különösen kellemes történés a harmadik és negyedik pályán, ahol a fél világ körbejárása után, közvetlenül a cél előtt van lehetőségünk arra, hogy a legfelső emeletről a legalsóra zuhanjunk vissza.

Ha a viszontagságok ellenére mégis sikerül eljutnunk a gazdit tartalmazó butéliához, a gép közli velünk, hogy éppen kit sikerült kiszabadítanunk (sorrendben:Velma, Fred, Daphne és Shaggy) és próbálkozhatunk a következő pályán. Ha Scooby-jaink elfogytak, közli velünk hány pontot értünk el és eredményünktől függően bekerülhetünk a HIGH SCORE TOP 10-be.

A SCOOBY DOO 1986 egyik legjobb grafikájú arcade-játéka volt, akik egyszer látták biztosan megkedvelték. Reméljük, a rajzfilmsorozattal nemsokára hazánkban is találkozhatnak szélesebb néprétegek is. Végül megjegyeznénk, hogy kiadványunk hollywood-i tudósítója lapzárta előtt néhány perccel jelentette, hogy a Hanna-Barbera producerkettős a Flinstone család és a Scooby Doo című sorozatait megpróbálja egy kalap alá vonni. A stúdiók mélyén, legnagyobb titokban folyó munkálatokról eddig mindössze annyi szivárgott ki, hogy egy Frédi nevű - bizonyára közismert - szereplő mindig azt mondja, hogy SCOOBYDOOBYDOO...

Agent X

A program file-térképe a következő: 314/210/7768/6339/10850/17350/13105/10900

Az örökélethez töltsük be a LOADER-t, RESET-eljük a gépet, és írjuk be a következő BASIC programot:

```
10 CLEAR 24999: LOAD"" CODE: FOR i=50000 TO 50017: READ a: POKE i,a:
   NEXT i: RANDOMIZE USR 50000
20 DATA 221,33,168,97,17,88,30,62,255,55,205,114,195,62,201,50,207,104
```

Futtassuk a programot (RUN+ENTER) és indítsuk el a magnót.

Uridium

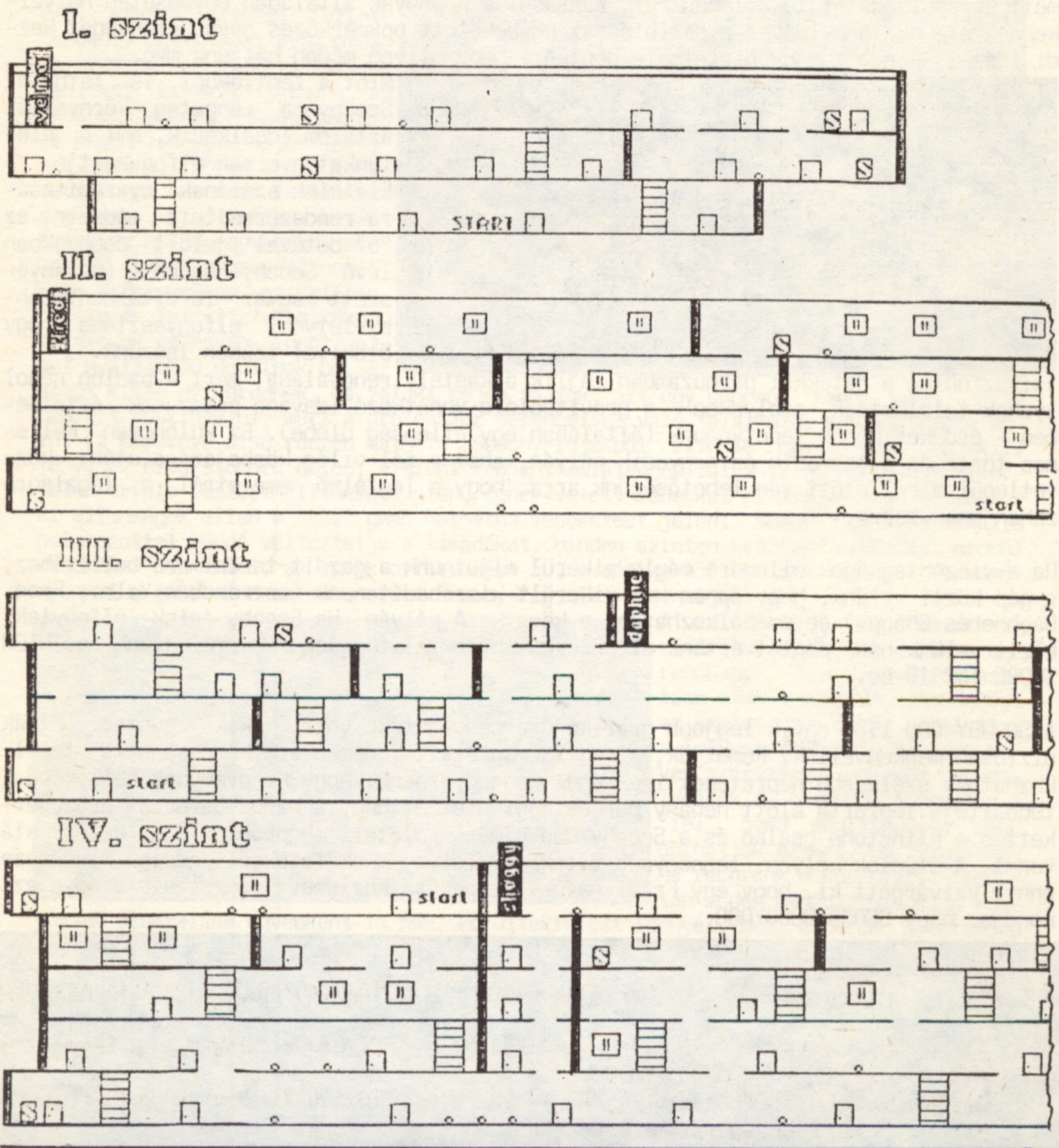
A program file-térképe a következő: 380/6912/38145

Az örökélet beviteléhez töltsük be a loader-t, és RESET-eljük a gépet. Írjuk be a következő BASIC programot:

```
10 CLEAR 27389: LOAD"" SCREEN$: LOAD"" CODE: POKE 31308,0:
   RANDOMIZE USR 64848
```

Futtassuk a programot (RUN+ENTER), és indítsuk a magnetofont.

SCOOBY DOO térkép



Super Robin Hood

A file-térkép: fejléces SCREEN és 40577 byte. A sérthetetlenséghez az 50516 címtől a következő byte-okat kell elhelyeznünk: 62,255,0,0,0,0,0, 0 és az 50532-es címre 255-öt.

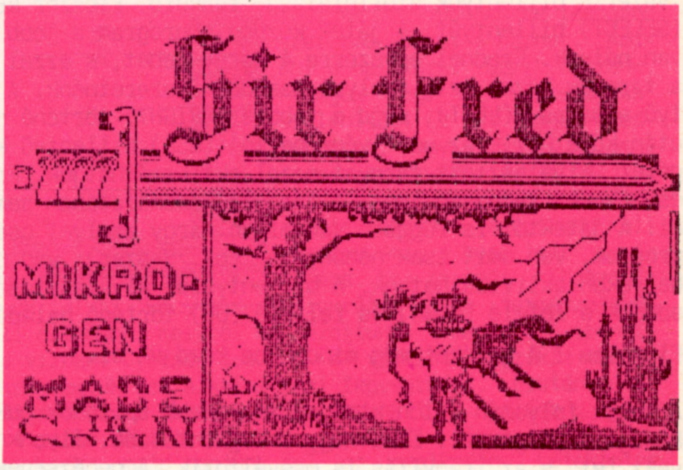
Töltsük be a LOADER-t majd RESET-eljük a gépet. Írjuk be a következő BASIC programot:

```
10 LOAD"" SCREEN$: FOR i=23296 TO 23340: READ a: POKE i,a: NEXT i
20 RANDOMIZE USR 23296
30 DATA 221,33,124,95,17,129,158,62,177,55,205,86,5,210,0,0,33,84,197,
   54,62,35,54,255,175,35,119,35,119,35,119,35,119,35,119,35,119,33,
   100,197,54,255,195,153.138
```

Futtassuk a programot (RUN+ENTER) és indítsuk a magnót.

2.2 SIR FRED

Equinox, Frost Byte, Stainless Steel, Battle of the Planets, ..., Sir Fred; többek között az itt felsorolt néhány játékprogram megjelenése indított el egy döntő változást a MIKRO-GEN cég történetében. Ez annyit jelent, hogy a cég letért a jól kitaposott ösvényről, és sorban forgalmazza az érdekesebbnél érdekesebb, de a régi szisztémától eltérő stílusú játékokat. Ezek, bár mind jó kivitelezésűek, nincs meg bennük az az eredetiség, ami pl. a 'Wally' játékok esetében tapasztalható volt. Mégis, az áttérés időszakában született néhány 'mutáns' műalkotás, amely egyedi darabnak számít a Spectrum játékok körében. Ilyenek pl. a Battle of the Planets, a következő részben bemutatandó Equinox és egy igen fantasztikus alkotás, a most ismertetésre kerülő Sir Fred is. A játék még magán hordja a 'Wally' játékok sajátosságait, de már óriási ügyesség szükséges hozzá.
Ugyanez volt a helyzet az Equinox esetében is. A programozók azt próbálták megakadályozni, hogy egy egyszerű leírással a programot ne lehessen elsőre végigjátszani. El is érték céljukat, mindezidáig - a programozókon kívül - nem sok embernek sikerült végigjátszani ('kilőni') a játékot. Itt meg kell említenünk, hogy a mi útmutatónk sem 'tudott' teljes lenni. Ez ugyanis egy egész könyv-anyagot felemésztene, de erről még később szólunk. Talán a cég is érezte, hogy túl nagy fába vágja a fejszéjét, amikor az ötletet kidolgozta, ezért egy spanyol programozói gárda közreműködésével hozták létre a programot. A program végül is elkészült, és megérdemel minden elismerést.



A játék első betöltésre nem tűnik igazi 'bombázónak'. Valóban, a hang és a grafika nem kiemelkedő, de nem is rossz. Színvonala teljesen egyenrangú a már ismert 'Wally' grafikával és hanggal, sőt színeiben talán felül is múlja ezeket. A Sir Fred maximálisan kihasználja a Spectrum grafikai és színkezelő lehetőségeit. Hősünk egész bonyolult mozdulatsorokat végez; lót-fut, vív, mászik, úszik, stb. Mindez természetesen tökéletes sprite-okkal, és folyamatos mozdulatokkal. Természetesen egy külső szemlélőnek a játék folyamatos végigjátszása szinte rajzfilmként hathat. E tökéletes felépítéshez már csak egy bájos történetet kellett keresni, amit a szerzők meg is találtak...
Hol volt, hol nem volt, volt egyszer egy király. Igazságos volt, népe nem ismerte az erőszakot, királyságában béke honolt és nyugalom. Történt egyszer, hogy a Feare kastély vastag falai között felbukkant egy gonosz fekete lovag, Sir Hugh D'unnyt. Boszorkányos tervet eszelt ki, megkaparintja a hercegnőt és a rabszolgájává teszi. Tervét véghez is vitte, s királyunk a békés csendet felverve kürtölte szét birodalmában: 'Ki küzd meg a hercegnőért ?'
Hivatta lovagjait, de sajnos Sir Vival, Sir Prize és Sir Spender éppen távoli földön tartózkodtak, megpróbálták kideríteni egy elvarázsolt páncéling sorsát.
Az egész birodalomban csak egy olyan lovag maradt, aki még nem volt túl a 90 éven. Nem sok látnivaló volt rajta. Rozsdás páncéljával a kezében, mindettől függetlenül szívében égő tűzzel, elindult a titokzatos veszélyek felé. Ő főhősünk, Sir Fred.

Itt kapcsolódunk be a történetbe, amely a továbbiakban abban a várban folytatódik, amelyben fogva tartja a fekete lovag a hercegnőt. A hatalmas erőd több mint 50 szobából áll. Az erőd egyes helyszínein, elsősorban az őrök feledékenységének köszönhetően különböző fegyverek, élelmiszer, stb. találhatók, melyeket Fred természetesen örömmel vesz magához.

2.2 SIR FRED

Küldetésünk mégsem 'leányálom'. Az erőd tele van csapdákkal, gonosz őrökkel. Frednek ellenfelét minimum háromszor kell megvágnia, hogy az elhalálozzon, kivéve, ha azt sarokba szorította. Ilyenkor egy vágás is elég.
Az erődben itt-ott elhelyezett tárgyakat úgy tekinthetjük, mint a színházi kellékeket, melyeket minden előadás után összeszednek, s korántsem biztos, hogy a következő előadáson ugyan abba a pozícióba helyezik el őket. Ez a játékban úgy jelentkezik, hogy ahol egyszer a kard volt, ott a következő játékban pl. gyufa, vagy esetleg íj lesz. A játék így bizonytalanná válik, aminek az lesz a következménye, hogy nagyon sokféleképpen végigjátszhatjuk a játékot, vagyis más tárgyakkal, más útvonalon, de mindenképpen a hercegnő a cél. A mi verziónk a "kard útja" nevet viselhetné, hiszen folyton jobbra-balra kell szurkálnunk, döfködnünk. Általában minden 5. állás hasonló ahhoz, mint amilyet mi itt ismertetünk.
Úgy gondoljuk megfelelő és alapos bevezetést írtunk a 'Sir Fred' nevű akció-kaland programhoz, hiszen megérdemli.

A játék betöltését követően megjelenik a vezérlési menü, melyben sorban a következő választásaink adódnak:

1-KEMPSTON ; 2-CURSOR ; 3-REDEFINE KEYS ; 4-KEYBOARD

KEMPSTON illesztő hiányában célszerű közvetlenül a 3. opciót kiválasztanunk, ez a 'vezérlő billentyűk átdfiniálása' funkció. Először kell kijelölnünk a KIVÁLASZTÁS (SELECT) billentyűt. A játék közben a program a jobb alsó sarokban jelzi, hogy mi van nálunk. Frednél egyszerre 4 tárgy lehet, s ezzel a billentyűvel választhatjuk ki a tárgyak közül azt, amelyet használni szeretnénk. A következő kiválasztandó billentyű a balra- (LEFT), majd a jobbra- (RIGHT) történő mozgás vezérlésére szolgál. Ezután következik az un. 'végrehajtó' (USE) billentyű kijelölése. A 'SELECT' segítségével kiválasztott tárgyat ezzel aktivizálhatjuk. Fred a kardon kívül íjat, nyilat és köveket is használhat. A tárgyak használatában sem tudunk egységes útmutatót adni. A játéktól is függően van olyan tárgy amelyet csak egyszer enged meg használni a program, van amit csak néhányszor (pl. 9-szer), van olyan is, amelyet csak bizonyos helyeken aktivizálhatunk. A billentyűzet kiválasztásának utolsó lépésében meg kell adnunk a lefelé (DOWN) és felfelé (UP) történő haladást vezérlő billentyűket.

Figyelem ! A játékot még mindig ne kezdjük el.

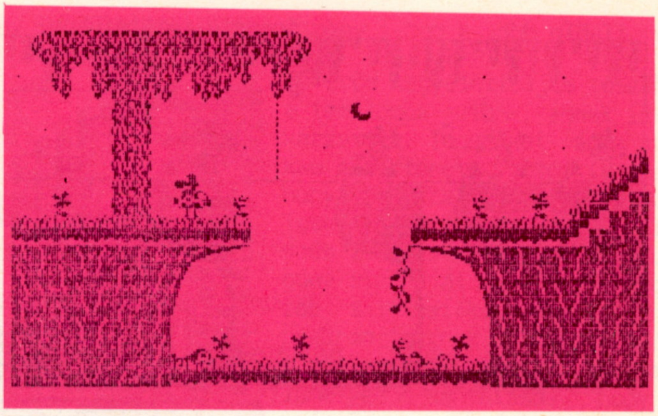
A játék folyamán külön kiragadott kisebb epizódok bukkannak elő, pl. úszás, vívás, kőhajítás, íjászat. Ez a négy a legfontosabb. Ilyenkor Fred egészen más pozíciókat vesz fel, s az irányítás is megváltozik. A nehézségek elkerülése végett megpróbáljuk röviden ismertetni a funkciók lényegét.

-Úszás: Már a második pályán meg kell tanulnunk úszni. Rövidebb gyakorlás után meglátjuk, menni is fog. Fred speciális vízalatti úszást végez. Ha nem irányítjuk, azonnal a felszínre tör oxigénért.

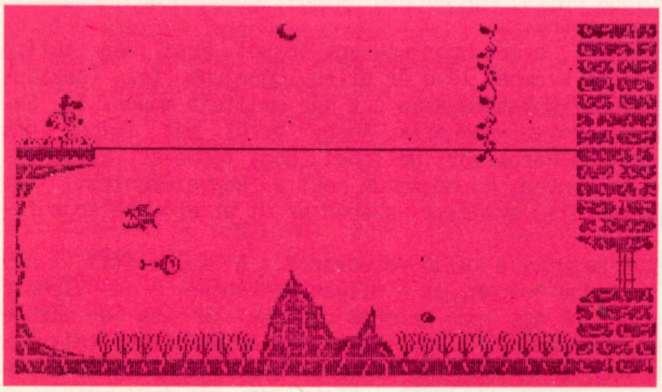
-Vívás: Mint tudjuk a víváshoz egy kard is szükségeltetik, így először ezt kell megszereznünk. Másodszor a kardozási funkció csak akkor lép üzembe, ha őrrel találkozunk. Ilyenkor aztán a fel/le billentyűk gyors nyomogatásával háríthatjuk a szúrásokat, míg az előre billentyűvel támadunk, a hátra billentyűvel pedig védekezve hátrálunk. Mindemellett ez a legnehezebb funkció, ezért célszerű, ha sokat gyakoroljuk.

-Íjászat: Ha felvettük az íjat, bármire lőhetünk, de nem minden és nem mindenki hal meg íjunk hatására. Használata esetén nyomjuk a lövés funkciót egyfolytában. Ekkor a kifeszített nyílvessző iránya 4 fokozatban elkezd változni, aszerint, hogy milyen magasra akarunk lőni. Ha kiválasztottuk az irányt, engedjük el a lövés-billentyűt, s a nyílvessző a kívánt irányba repül. Egy íjhoz 9 db. nyílvessző tartozik.

2.2 SIR FRED



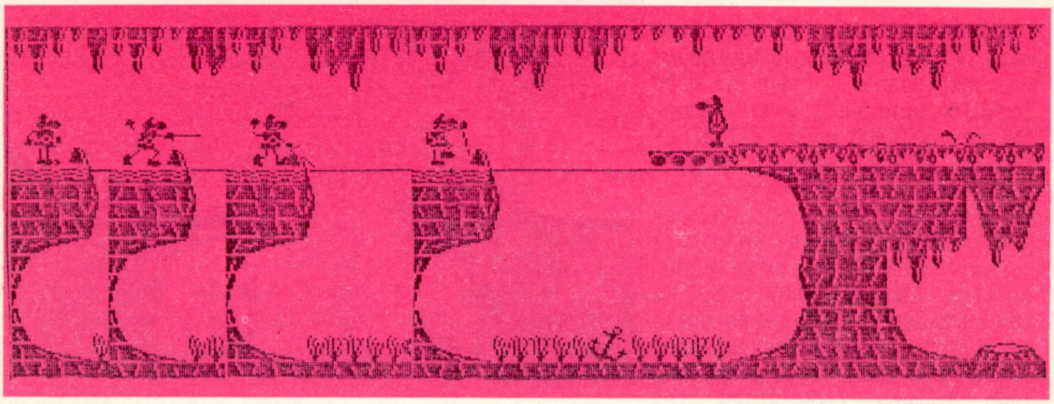
Ennyi előzetes információval már nekikezdhetünk küldetésünk végrehajtásának. Fred az erőd legkülső kertjéből indul. Azonnal egy mélyebb gödör állja utunkat, s külön szivola, hogy a gödör mélyén egy telepített vipera fickándozik. Óvatosan menjünk a gödör szélére, majd amikor a vipera elvánszorog a jobb oldalra, lépjünk le. Fred, ha nem is talpra, de fenékre esik, így kisebb energiaveszteséggel megússza a dolgot. A gödör mélyén örömmel fedezhetünk fel egy jókora csirkecombot. Vegyük fel azt, majd amikor a vipera elindul visszafelé, kövessük és ugorjunk fel a lelógó indára. Sikerült kijutnunk a veremből, átmehetünk a következő pályára. Az erődöt hármas védelmi rendszer védi: az erdő, egy veremrendszer és egy vizesárok. Most ezek közül a legutolsóhoz jutottunk.



Ez a hely gyakran bővelkedik órmótlan vizi szörnyekben. Ebben az esetben egy magányos piranha 'nyüzsög' a vízben. Mit tegyünk ? ...Dobjuk be legújabb szerzeményünket a halnak. Amikor a hal közel úszik hozzánk, nyomjuk meg a 'végrehajtás' billentyűt. Ez azt fogja eredményezni, hogy a 'sült galamb' lassan ereszkedve a hal szájába esik. Ő azonban csak a fenéken tud lakmározni, ezért meg kell várnia, amíg a husi leér a víz aljára. Ezt az időt kihasználva, toccsanjunk a vízbe és vegyük fel a 'kecskebéka figurát'. Ezen természetesen Fred speciális úszás-módját kell értenünk, melyről már szóltunk. A megfelelő irányítással gyorsan előrejuthatunk a vízben. Gyorsan ússzunk a vízalatti szikla mögé, ahol kilenc kődarab hever. Ezeket vegyük fel alkalmi kézi fegyverként. Ezután ússzunk a jobboldali, vízalatti átjáróhoz és tuszkoljuk be magunkat a nyíláson. Fred éppen, hogy átfér a földalatti átjárón. A túloldalon a vízalatti tárna található. Óvatosan másszunk ki a szikla peremére, majd menjünk fel az emelkedőn. Letekintve egy polipot láthatunk, amint le/fel úszkál a vízinövények között. A szikla vízalatti beugrójában azonban, egy valódi acélélű kard hever. Ezt mindenáron meg kell szereznünk. A polip azonban komoly akadályt jelent, de van egy megoldás...essünk le, amikor a polip éppen lent van, és másszunk ki a tárna jobb szélén, a kis vízesésen. Használjuk az imént szerzett kavicsokat, a már említett módon. Érdemes kicsit dobni, így a kő beleütközik egy lelógó cseppkőbe, erről lepattanva pedig, egyenesen áldozatunk fejére, aki így nagyon gyorsan elhalálozik. Most már minden nehézség nélkül felvehetjük a kardot a kövek mellé. Kimászva a vízből induljunk jobbra. Ez a tárnaátkelőhely, az erőd titkos kereskedelmi útvonala. Ezen a tárnacsarnokon hozzák be az élelmiszert a várba. Mint látjuk a vízen történő átkelést egy tutaj biztosítja, amit révész kezel. A vigyori révész eléggé idült alkoholos állapotban van, ezért nem is hajlandó minket átvinni. Ekkor viszont észrevesszük az orrunk előtt heverő alattomos kődarabot, melyet természetesen nem tudunk átugrani. Hogyan juthatunk át ?

A kő egy titkos csapóajtót rejt magában, amely a révész utáni pályára, vagyis az 1. számú vízalatti őrhelyre vezet. Ezen a csapóajtón bonyolítják le az őrök a váltást. A csapóajtó azonban csak a megfelelő 'jelszóra' nyílik ki, amely jelen esetben egy vívó mozdulatsort jelent. Tegyük tehát a következőket: hátráljunk (balra), amíg éppen vissza nem lépünk az előző pályára (tárna). Ezután forduljunk a kő felé, vigyük

2.2 SIR FRED

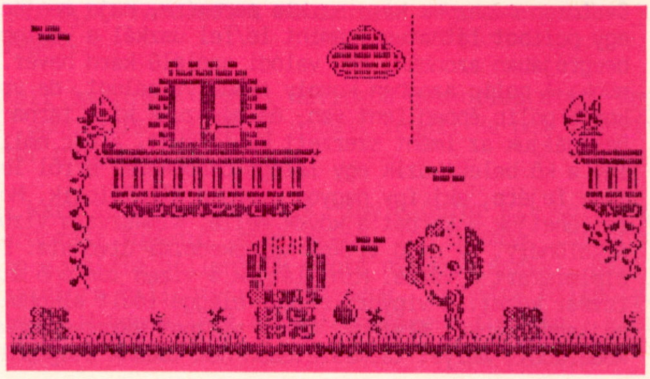


a kurzort a kardra és nyomjuk meg a lövést. Ennek az lesz a hatása, hogy Fred egy szúrómozdulatot tesz a kő felé. Ezt követően nyomjuk meg az előre/jobbra billentyűt. Most lovagunk egy lépést tett előre, kardjával lefelé mutatva. Látszólag nem történik semmi. Ha azonban most kimegyünk balra, még mindig karddal a kezünkben, meglepve láthatjuk, hogy az 1.sz. őrhelyen vagyunk.

Fejünk éppen, hogy kilátszik a földből, ezért ugorjunk egy nagyot. Most már tettre készen indulhatunk támadásra a velünk szemben lévő őr ellen. Ugorjunk fel (nekifutásból) a padkára, s kardunkkal lendüljünk támadásba (a már említett módon). Ekkorra már ellenfelünk sem rest és keményen védekezik. Ahogy arról már írtunk nem elég csak egyszer beleszúrni, legalább háromszor, de lehet, hogy többször kell elvégeznünk ezt a cselekvést, mire kiszenved. Ha megadtuk neki a kegyelemdöfést, tulajdonképpen elhárítottuk az összes külső akadályt, szabad az út az erőd belseje felé.

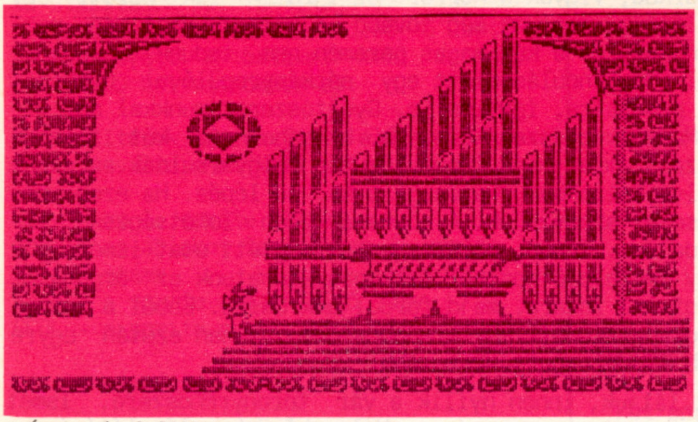
Következő célunk a kúton keresztül bejutni a külső várudvarra. A földalatti 1.sz. őrhelyről egyenesen a kút víztárolójába jutunk, és - milyen szerencse - a vízhúzó kötél pont le van eresztve. Mint látjuk egy igen értékes portéka is van a vízben (ahogy az már lenni szokott), ami nem más, mint egy íj, tíz darab nyílvesszővel. Valakinek azonban az a paranoiája, hogy ahol víz van, ott piranhákat telepít. Ennek a nyoma sajnos itt is megmutatkozik. A kút vizében ott lubickol egy nem éppen növényevő példány. A hal azonban néha megszédül a körbe-körbe való úszástól és kis időre megdermed. Ezt az időt kihasználva ússzunk le az íjért, majd másszunk fel a vízhúzó kötélen.

A várudvar kellős közepén vagyunk. Legyünk óvatosak, mert már a belső területen vagyunk, itt pedig hemzsegnek az őrök (karddal) és a mesterlövészek (íjjal).

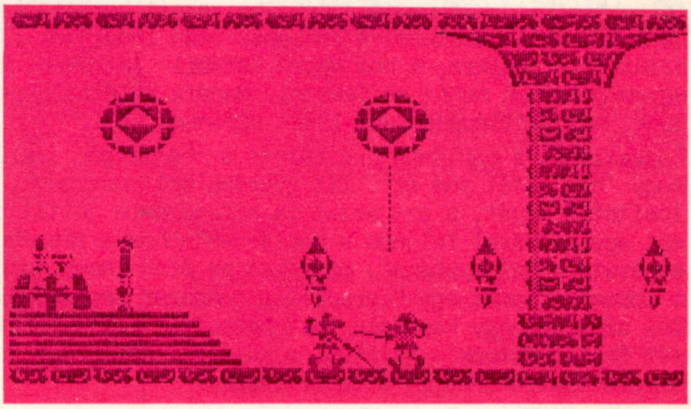


A díszes erkélyekről liánok, indák lógnak. Az egyik erkélyen egy íjász leselkedik, ezért legyünk készenlétben (vigyük a kurzort az íjra). A bal szélső folyondár alkalmasnak látszik, hogy felmásszunk rajta. Felérve célozzunk, és lássuk el íjászbarátunkat egy halálos nyílvesszővel. Mint látjuk az erőd felső régióiból egy kötél lóg lefelé. Fussunk neki és kapaszkodjunk meg a kötélen. Természetesen ugrani kell. Ezután forduljunk át a kötél másik oldalára és várjuk meg, amíg az kileng. Felmászva, a virágcsarnokba jutottunk. Fred most egy eléggé kaszkadőr mutatványt készül végrehajtani. A virágcserépen állva át kellene jutnunk, a jobb oldali beugróra. Elvben nekifutunk a cserép széléről és ugrunk. Ennek a hipp-hopp megoldásnak azonban az lenne az eredménye, hogy visszazuhanunk a várudvarra. Tegyük tehát a következőt: Hátráljunk a virágcserép legszélére (balra), majd rohanjunk előre (nyomjuk egyfolytában a jobbra billentyűt). Elvileg le

kellene zuhannunk, a rohanás nagy lendületétől viszont nagy ívben a jobb oldali beugróra esünk (ismét fenékre). Innen már jobbra mehetünk.



Az erőd kegyeletteljes részébe, a templomba jutunk, annak is a karzatára. Innen két ágra szakad a történet, attól függően, hogy mekkora energiánk van. Több lehetőségünk adódik. Először is nézzük meg, hogy az orgona sípján nincs-e egy jó darab sült csirke. Ide ugyanis gyakran tesznek csirkét a kántor részére, aki olykor úgy belemélyed a kántálásba, hogy az evésről is megfeledkezik. Ha van csirke, álljunk a lépcső legfelső fokára, ugorjunk, majd másszunk fel érte az orgonasípokon. Azonnal el is használhatjuk.
Itt kell megemlítenünk, hogy az energiánk pótlására szolgáló csirkéket, a gép RND függvény szerint adja be, ezért már a start pályától figyeljünk minden pályát, hátha akad csirke valahol. Ezt a későbbiek számára tartalékolhatjuk.



Ha nincs szükségünk energiára vagy már feltöltöttük magunkat, beléphetünk a misézó csarnokba. Mint láthatjuk, a szószék mögött, illetve az áldozó asztal alatt vagyunk. A templomot egy igen 'kemény' őr védi, az eretnek vandálok ellen. Hiába próbálnánk bemagyarázni neki, hogy mi is keresztények vagyunk, biztos nem hinné el, ezért kénytelenek vagyunk eltávolítani. Egy óriási ugrással pattanjunk fel az áldozó asztalra, és kardunkkal rohanjuk le ellenfelünket. Ha támadásunk lesújtó, hamar (kb. 10 szúrásból) sarokba szoríthatjuk ellenfelünket, de vigyázzunk, mert ő is 'keni' kegyetlenül.
"Mi itt sem voltunk" jelszóval, angolosan távozzunk a helyszínről (jobbra). Mint látjuk (illetve nem látjuk) ezen a pályán sem enni, inni vagy ölni-való nincs. A padlóba azonban titkos billenő ajtó lett beépítve. Ezt a földbe rejtett kis kallantyú elfordításával hozhatjuk működésbe. Mint tapasztalni fogjuk, a padló valami antigravitációs berendezés hatására felemelkedik.
Visszatérve a csapóajtóra, várjuk meg, amíg az teljesen felnyílik, és jobbról balra haladva lépjünk le a keletkezett lyukba. A várudvar lépcsőkorlátjára esünk. Ha most jobbról/balra léptünk volna, gyors tempóban pattognánk végig a lépcsőkorláton. Ez nem lenne egészséges dolog, mivel Fred nem fizeti a kóbor lovagok biztosítási kötvényének havi esedékes részleteit, ezért küldetésünk nagy valószínűséggel véget érne. Így azonban felkapaszkodhatunk a korlát emelkedőjén. Fred itt telepatikusan megérzi, hogy a következő pályán 'kellemes meglepetés' várja, ezért fel is készül a harcra. Vigyük a kurzort a kardra, és hátráljunk a korlát egyenes részéig (jobbra). Mielőtt cselekednénk, elmagyarázzuk tervünk lényegét. A balra lévő pályán a korlát vége található, amelyre egy fontos tárgy, a gyújtószerszám lett elhelyezve. Valószínűleg az ide állított őr hagyta itt, két pipafüst között. Egyelőre azonban ott van az őr. Ha közömbösen átsétálnánk a következő pályára, annyi időnk sem maradna, hogy 'Jó napot'-ot szóljunk. Taktikánk a következő: a korlát legszéléről futunk be egyenesen az őr karjába, jobban mondva a kardjába. De mi sem vagyunk restek, rántsuk elő kardunkat és a szokásos módon lendüljünk támadásba. Próbáljuk minél jobban balra szorítani

ellenfelünket; így hamar végzünk vele. Dolgunk végeztével vegyük az irányt balra. Mielőtt átmennénk a szökőkutas parkba, vigyük a kurzort az íjra. A fák között egy sunyi mesterlövész bujkál. Ha kilőttük, menjünk még tovább, a kúthoz. Most már megindulhatnánk felfelé, a királylányhoz, de a kút magas peremén nem tudunk átmászni. Fred azonban most is magára talál. Meg indulás előtt egy sárkányrepülővel bombákat helyezett el az erőd 'vaskosabb' pontjaira, így ide is. Gyújtószerszámunkkal aktivizálhatjuk a robbanószerkezetet. Menjünk egészen közel a bombához (a kút tövében van), vigyük a kurzort a gyújtószerszámra, majd lövés. Ezután felgyorsulnak az események. A bomba pár másodperc múlva felrobban, hatósugara kb. 10 lépés. Ha ezen belül vagyunk, komoly égési sérüléseket szenvedünk. A robbanás után lehetségessé válik, hogy átmásszunk a kút túloldalára. Innen, a már egyszer megtett utat ismételjük meg, fel egészen az orgonáig. Egy megfelelő pontról felkapaszkodhatunk az orgonasípokra. Ehhez menjünk a lépcső legfelső fokára, majd forduljunk balra. Ebből a pozícióból ugorjunk, s Fred máris megkapaszkodott egy alsóbb sípban. Innen ezután egyre feljebb mászhatunk. A legmagasabb sípról ugorjunk tovább.

Az ötödik szintre kerülünk. Mint látjuk, innen nyílik a vár toronyrendszere. Ismét akadályba ütközünk, a padlásajtó ugyanis be van zárva. Hála hősünk előrelátásának, itt is egy bomba meggyújtásával oldjuk meg a problémát.

Ahhoz, hogy a padlásajtóhoz eljussunk, fel kell másznunk az erőd zászlójára. Fred szerencsére nem nehézsúlyú lovag, ezért a zászló nem szakad le alatta. Innen már felrobbanthatjuk a padlásajtóba helyezett bombát. Ahogy meggyullad a bomba, álljunk ki a padlás legszélére. A robbanás után induljunk a vár legmagasabb régióiba. Egyik szintről a másikra, a nem éppen stabil létrákon keresztül juthatunk át. Ezeken egyébként igen nehéz közlekedni. Magára a létrafokra egy ugrással felkapaszkodhatunk, majd kétlépésenként ismét ugranunk kell.

Közvetlenül a padlásbejáró felett a 'hajítógép' található, igazán nem mondhatjuk rá, hogy 'hajítófát sem ér'. Itt gyorsan kell másznunk, mivel a kőhajító szerkezet önállósította magát, s ezt azzal is kifejezésre juttatja, hogy kőzáport zúdít ránk. Ismét egy pályával feljebb a padlásnak ahhoz a részéhez érkeztünk, amely egyben titkos kijárata a hálószobának, másrészt az őrök innen jutnak fel a toronyállásokba. Magába a toronyba ismét egy hidraulikus rendszer segítségével juthatunk. Mint látjuk, a fal bal oldalába egy kapcsoló lett beépítve. Ezt elfordítva, lejön egy palló, s erről már felmászhatunk a kötélre, a kötélről pedig egy felhőre. A kapcsolót egyszerűen elfordíthatjuk, ha íjjal belelövünk.

Miután feljutottunk a toronyba, egyre jobban érezhetjük a hideg esős angliai időjárás lehelletét. Ez abban is megmutatkozik, hogy a vár körül dundi esőfelhők lebegnek. Mivel Fred a pehelysúlyú kóborlovagok kategóriájába sorolható, könnyen megáll a felhőn, sőt még a jég hátán is. Végre fellélegezhetünk, a dolog nehezén már túl vagyunk. A friss levegő új erőt önt belénk. A felhőről egy ugrással kapjuk el a toronyról lelógó kötelet, ezen pedig másszunk fel a torony csúcsára.

Fred nem tériszonyos, ezért bátran közlekedik az 'instabil' cserepeken. Van még egy akadály, amit Fred már idejöttekor meglátott. A balszélső bástyán egy hajítógép és egy íjász téblábol. Ezeket kell még lekezelnünk. Menjünk tehát balra két pályát, de előtte helyezzük készenlétbe az íjat. Miután átmentünk a hajítógépes pályán, eljön számunkra az utolsó pálya. Ahogy beléptünk, célozzuk meg az íjászt, majd eresszünk belé egy halált hozó nyílvesszőt.

És a mese itt véget ér, ott a gyönyörű hercegnő. Az események innen már maguktól mennek...csók...és egy kis meglepetés.

Figyelem! A Sir Fred bármennyire is tökéletes, az adatkezelésébe menet közben hiba csúszhat, ezért ha a játék végén az utolsó pályán mégsem lenne ott a hercegnő, ne keseredjünk el, ilyenkor ugyanis a gép 'elfelejtette' kirakni a hercegnő sprite-ját. Mindenkit érhetnek csalódások, ám a játék így is véget ér, de sajnos a puszi elmarad.

Memóriamozgatás sprite és egy másik sprite-ablak között:

Az utasítások szintaktikája az előzőekkel azonos, csak itt az utolsó 'S' betűjelek helyett 'M' betűket kell írnunk, pl. GWBLS helyett GWBLM-et.

Az utasítások végrehajtása előtt a következő változókat kell definiálnunk:

SP1 - az 1. sprite száma
SP2 - az ablakot tartalmazó 2. sprite száma
SROW - a sprite ablak felső sora
SCOL - a sprite ablak baloldali oszlopa

Másoló utasítások:

Két sprite közötti adatátvitelre adnak lehetőséget. A végrehajtás előtt a következő változókat kell definiálnunk:

SP1 - az 1. sprite száma
SP2 - a 2. sprite száma

COPYM - SP1 átmásolása SP2-be
COPORM - SP1 ráillesztése OR-ral SP2-re
COPXRM - SP1 ráillesztése XOR-ral SP2-re
COPNDM - SP1 ráillesztése AND-del SP2-re
COPATTM- SP1 attributumainak átmásolása SP2-be

Megjegyzés: Ha a két sprite mérete különbözik, nem történik semmi.

TRANSZFORMÁCIÓK

A transzformációkhoz tartoznak: a forgatás, tükrözés, invertálás és a nagyítás.

Invertálás:

Az attributum adatok változatlan állapota mellett minden 1 értékű képpont 0 lesz és fordítva.

INVM - a megadott számú sprite invertálása (SPN-ben előzetesen be kell állítanunk a sprite számát)
INVV - meghatározott méretű ablak invertálása (az ablak adatait a COL, ROW, LEN és a HGT változókban be kell állítanunk)

Tükrözés:

MIRM - a sprite képpont adatait tükrözi középpontra vetítve
MARM - a sprite attributumait tükrözi középpontra vetítve (a sprite számát előzetesen az SPN változóban be kell állítani)
MIRV - meghatározott méretű ablak képpont adatait középpontra tükrözi
MARV - meghatározott méretű ablak attributumait középpontra tükrözi (az ablak adatait a COL, ROW, LEN ill. a HGT változókban be kell állítani)

Forgatás:

Két sprite között adatforgatást végez el. Ennek az az alapfeltétele, hogy a sprite elforgatott méretei megegyezzenek. Pl. ha egy 5x2-es méretű sprite-ot akarunk elforgatni (SP2), akkor SP1-nek 2x5 méretűnek kell lennie. Sprite önmagába nem forgatható, tehát SP1 nem lehet azonos SP-vel.

SPINM - SP1 tartalmát SP2-be forgatja 90 fokkal (SP1 és SP2 értékét előzetesen be kell állítanunk)

Nagyítás:

A forgatáshoz hasonlóan a kiinduló sprite számát itt is SP2-ben kell beállítanunk, és az utasítás végrehajtása után az eredmény SP1-ben jelenik meg. SP1-nek pontosan kétszer nagyobb méretűnek (tehát négyszeresnek) kell lennie, mint az SP2 sprite.

DSPM - SP2 nagyítása az SP1 sprite-ba

Megjegyzés: ez utóbbi két transzformáció csak a sprite memóriában hatásos

MEGSZAKÍTÁSOKKAL ÖSSZEFÜGGŐ SZAVAK

Ezek az utasítások ellenőrzik a White Lightning szavak "Előtér/háttér " végrehajtását.

HALT - felfüggeszti a CPU működését a következő megszakításig

EI - engedélyezi a megszakításokat

DI - tiltja a megszakításokat

EXX - az IDEAL változóit kicseréli a váltakozó IDEAL változókra. Ha megszakítás van, ez automatikusan végrehajtódik, megszakítás után pedig visszaállnak az előtér változók. Ha a háttérprogramot nem használjuk, elsősorban plusz változókat kapunk.

INT-ON - minden megszakításkor végrehajtódik a megadott szó.
Pl. VONAL INT-ON beállítja a VONAL szót, amely a háttérben működik.

INT-OFF- visszatérést jelent az 1. megszakítási módba. A háttérprogram futása befejeződik.

BASIC

INTERFACE SZAVAK

Igen hatékony lehetőség, ugyanis ezeknek az utasításoknak a segítségével keverni lehet a programunkon belül a BASIC-et és a FORTH-t.

PROG - áttérés a BASIC rendszerre

RESERVE- helyet foglal a szótárban a BASIC részére
Pl. 2048 RESERVE (+ENTER)
2 Kbyte-ot foglal le

GOTO - a program futása a TOS-ban meghatározott BASIC soron folytatódik
BASIC-ből a RANDOMIZE USR 30000 utasítás segítségével indíthatjuk újra a FORTH-t, míg ha csak vissza szeretnénk lépni a FORTH azon pontjára, ahonnét leágaztunk, úgy RANDOMIZE USR 30006 a megfelelő lépés.
Pl. 120 GOTO (+ENTER)
a 120-as BASIC sorra ugrik át

RETUSR - a programvégrehajtás a FORTH-t hívó RANDOMIZE USR 30000 utáni utasításra ugrik

KÜLÖNFÉLE SZAVAK

SETAV - meghatározott (COL, ROW, HGT és LEN) ablak attributumait beállítja az aktuális INK és PAPER színekre.

SETAM - az SPN-nel meghatározott sprite attributumaiba az aktuális INK és PAPER színeket viszi be

CLSV - az aktuális képernyő-ablak törlése

CLSM - törli az SPN-nel meghatározott sprite képpont adatait

ADJM - a PUT/GET utasítások 2. csoportjával együtt felhasználva hatékony. Ha a COL, ROW, LEN és HGT által meghatározott sprite kilógna a képernyőn, úgy a méretadatok megfelelően módosulnak, és az ablak ill. a sprite csak részben kerül a képernyőre.

ADJV - alapjában azonos az előzővel, csak nem a sprite-ra, hanem egy ablakra vonatkoztatva

SCANM - ütközések vizsgálatánál jelentős, megvizsgálja, hogy a sprite belsejében van-e kigyújtott képpont. Valós esetben egyes kerül a verem tetejére, máskülönben zérus.

SCANV - az előzőhöz hasonló, de ez a COL és ROW szerint meghatározott karakter-pozícióról ad információt

RND - a TOS-t egy 0 és TOS közötti véletlen számmal váltja fel

OUT - TOS2 tartalmát a TOS-ban meghatározott című portra küldi

IN - felváltja a TOS-beli port címet az onnan olvasott 8 bites szám 16 bites megfelelőjével

BLEEP - hanghatás, melyhez TOS adja a hangmagasságot, TOS2 pedig az időtartamot

ATTON - ha végrehajtuk ezt az utasítást, ezután a PUT/GET 1. csoport utasításai érvényesek lesznek az attributumok mozgatására is

CALL - ugrás a TOS szerinti címen meghatározott gépi kódú rutinra

ZAP - programírás végén ezzel áll elő a végleges verzió. A gép kiírja a program hosszát, majd az elmenthető a SAVE"név" CODE 24830, HOSSZ utasítással, ahol HOSSZ az előzetesen megjelent érték.

ZAPINT- olyan programok esetében, amelyek használják az "Előtér/háttér" műveletet ZAP helyett a ZAPINT utasítás használata javasolt

PRT-ON- a további output a ZX-nyomtatóra kerül

PRT-OFF-további output a képernyőre

BASIC A FORTH-BAN

A BASIC-ből is ismert parancsokhoz tartozó paramétereknek a veremben felülről lefelé, megfelelő (a BASIC-kel azonos) sorrendben kell elhelyezkedniük. Hiba esetén a FORTH-ba visszaléphetünk a meleg-indítással: PRINT USR 2483.
A paramétereket zárójelek közé helyeztük.

COPY	- nyomtatás a ZX nyomtatóra
AT	- kiíratási pozíció beállítása (ROW, COL)
BORDER	- keretszín beállítása (színkód)
CLS	- képpont-adatok törlése, aktuális attributumok beállítása
DRAW-ARC	- ív rajzolása az aktuális pontból (X,Y,szög)
CIRCLE	- kör rajzolása (X,Y,sugár)
DRAW	- vonal rajzolása egy meghatározott pontból (X,Y)
PLOT	- pont kijelölése a képernyőn (X,Y)
ATTR	- kijelzi a megadott karakter attributum kódját (ROW, COL)
POINT	- a megadott képpont állapotát kérdezi le. Kigyújtott állapotban a verem tetejére egyes kerül. (X,Y)
TAB	- a kiírás oszloppozíciójának beállítása (COL)
INK	- a tintaszín beállítása (színkód)
PAPER	- a háttérszín beállítása (színkód)
OVER	- felülírás üzemmód beállítása (1/0)
INVERSE	- inverz üzemmód beállítása (1/0)
BRIGHT	- fényesség beállítása (1/0)

ELŐTÉR-HÁTTÉR

Játékprogramok tervezésekor gyakran ütközhetünk abba a problémába, hogy egyes műveleteket azonos időközönként kell végrehajtanunk, ezek időzítése viszont a közvetlen folyamatban igen nehézkes. Sok számítást és elpazarolt processzor-időt jelenthet. A megoldás kulcsa a kettes megszakítási rendszer használata, vagyis az, hogy kettes megszakítási módba kapcsolva minden másodpercben 50-szer végrehajtunk egy meghatározott rutint. A kettes megszakítás kezelése gépi kódú oldalról sem nehéz, de a gyakorlatlan programozó számára gondot okozhat. A White Lightning-ben az INT-ON és az INT-OFF parancsok segítségével lehetővé válik a kettes megszakítási rendszer kezelése. Most az ezzel kapcsolatos ismereteket tekintjük át.

A megszakítás bekövetkeztekor az előtér-program futása megáll, és a háttér-program kerül végrehajtásra. Ezt elvégezve visszatér a program az előtér programhoz.
A háttér-programot tehát 1/50-ed másodpercen belül kell tudnia lefuttatnia, hogy annak befejezését követően vissza is tudjon térni. Nem jelent problémát viszont az sem, ha a háttér-program futási ideje meghaladja az 1/50 másodpercet. Ekkor nem 50-szer, hanem csak 25-ször lesz végrehajtva másodpercenként. Ha az 1/25-öd másodpercet is meghaladná, akkor már csak 13-szor.
A háttér-program jelenléte gyakran problémát okozhat. Nézzünk erre egy gyakori hibalehetőséget. A háttér-program a képernyőn egy folyamatos scroll-t hajt végre, míg az előtér-program éppen karakterek meghatározott sorozatát helyezi ki a képernyőre. A gond akkor jelentkezik, amikor a karakterek megjelenítése közben jön a megszakítás, és még csak a karakterek fele lett elhelyezve a képernyőn. Végrehajtódik a scroll, a karakterek fele arrébb csúszik, majd megjelennek a további karakterek is. Igen ám, de a közben jelentkező scroll miatt elcsúszás lépett fel a karakterek között. Ezt a hibajelenséget kiküszöbölhetjük, ha az ilyen fontos és megszakításmentes előtér-programok végrehajtási idejére nem engedélyezzük a megszakításokat, azaz a karakterek kiíratása előtt kiadunk egy DI utána pedig egy EI parancsot.

A háttér-program futtatásának szintaktikája a következő:

'NÉV INT-ON (+ENTER)

azaz aposztróf - háttér-program neve - space - INT-ON - és végül ENTER

Ha a háttér-program a végrehajtáshoz szükséges változókat saját maga nem definiálja, úgy azokat nekünk kell definiálnunk.

Pl. a képernyő közepén meghatározott méretű (5x5) ablakot egy háttér-program segítségével scrollozzunk és invertáljunk.
Mindenekelőtt definiálnunk kell egy szót, amelyre hivatkozni fogunk a háttér-program meghívásakor:

```
Pl. :HPRG WRR1V INVV;
```

Most állítsuk be a paramétereket is:

```
5 HGT' ! 5 LEN' ! 13 COL' ! 8 ROW' !
(+ENTER)
```

Írassunk ki tetszőleges adatokat a képernyőre:

```
VLIST (+ENTER)
```

Végül adjuk ki:

```
'HPRG INT-ON (+ENTER)
```

Azt fogjuk látni, hogy egy kicsit gyors a futási eredmény. A háttér-program megállítható az INT-OFF segítségével.

A megszakításokat lelassíthatjuk egy másik programmal, pl. legyen megszakítás 1/10 másodpercenként. Először is ismét definiálunk egy új változót és egy új szót:

Az új változó: 0 VARIABLE VALT (+ENTER)

Az új szó: :HPRGA VALT (0) 1+5 IF HPRG
0 VALT / ELSE 1 VALT +1 ENDIF

Ezután írjuk be: 'HPRGA INT-ON (+ENTER)

FREKVENCIA ÉS FÁZIS

A háttér-programok kezelésének, illetve a megszakítás-vezérlésnek az egyik és fő problémája, hogy egy esetleges pontvizsgálat megelőzheti az ugyanazon pontbeli képernyő-műveletet. Ennek következtében a képernyőn furcsa dolgoknak lehetünk tanúi, a képek egyes részletei úgy néznek ki, mintha felszeletelték volna azokat.
Tételezzük fel, hogy van 4 különböző szó, melyeket 4 különböző frekvenciával akarunk végrehajtani. Legyenek ezek a következők: INVV minden másodpercben 50-szer, MIRV minden másodpercben 20-szor, a WRR1V minden másodpercben 4-szer és végül a WCRV másodpercenként 5-ször.
Ezt követően definiálunk egy változót, mely számolni fogja a megszakításokat és 4 konstanst a frekvenciák tárolására.

```
0 VARIABLE VALT 50 CONSTANT K1 20
CONSTANT K2 4 CONSTANT K3 5 CONSTANT
K4 (+ENTER)
:MAO MOD ABS 0= ; (+ENTER)
:IRUN VALT @ DUP DUP DUP (+ENTER)
K1 MAO IF INVV ENDIF (+ENTER)
K2 MAO IF MIRV ENDIF (+ENTER)
K3 MAO IF WRR1V ENDIF (+ENTER)
K4 MAO IF WCRV ENDIF (+ENTER)
1 VALT @ +1 ; (+ENTER)
```

Ezután csak annyit kell tennünk, hogy begépeljük a paramétereket:

```
10 COL' ! 10 ROW' ! 6 HGT' ! 8 LEN' !
2 NPX' ! (+ENTER)
```

Vigyünk ki adatokat a képernyőre:

```
VLIST 'IRUN INT-ON (+ENTER)
```

INT-OFF segítségével kiléphetünk a háttér-programból.

Sokszor nem elég a frekvenciák ellenőrzése, célszerű bevezetni a fázis-formációkat. Ebben az esetben további négy változóra is szükségünk lesz. Először is adjuk ki: FORGET IRUN, majd gépeljük be a következőket:

```
31 CONSTANT KH1 5 CONSTANT KH2 0
CONSTANT KH3 3 CONSTANT KH4 (+ENTER)
```

Ezután írjuk be:

```
:IRUN VALT @ DUP DUP DUP (+ENTER)
K1 MOD ABS KH1= IF INVV ENDIF(+ENTER)
K2 MOD ABS KH2 = IF MIRV ENDIF
(+ENTER)
K3 MOD ABS KH3 = IF WRR1V ENDIF
(+ENTER)
K4 MOD ABS KH4 = IF WCRV ENDIF
(+ENTER)
1 VALT +1 ; (+ENTER)
```

Végül hajtsuk végre a programot:
VLIST 'IRUN INT-ON (+ENTER)
A megállítása INT-OFF-al, törlése FORGET IRUN-nal lehetséges.

4. VIEWPOINT

Az ACS software által forgalmazott, és 1982-ben megelent program úttörő Spectrumon a 3D grafikai technika vonatkozásában. Egyszerű felépítése, érthető kezelése tette népszerűvé, ugyanakkor felépítése 100 % BASIC, így igazán látványos eredményeket ne várjunk tőle, viszont kilistázhatjuk a programot, és kifejthetjük a működési mechanizmust.

A program terjedelme: 6685 byte és egyszerűen betölthető a

LOAD "viewpoint" vagy a
LOAD "" parancsokkal.

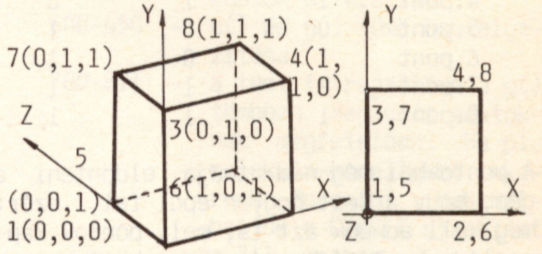
A Viewpoint automatikusan indul, és egy menüvel jelentkezik be:

Press: 1 to input new co-ordinates etc.
(adatbevitel a billentyűzetről)
2 for rotation about a point
(forgatás a három dimenzió bármely pontja körül)
3 for rotation about a line between two points
(forgatás egy két pont által határolt egyenes körül)
4 for view from one point to another
(forgatás megadott két pont relatív helyzete szerint ld.később)
5 for view of starting position from x,y plane
(a mindenkori x,y koordináta kép megtekintése)
6 to save details
(adatállomány kimentése magnetofonra, adattömb formájában)
7 to load new details
(új adatállomány betöltése a magnetofonról)

Az adatok bevitelét kétféleképpen oldhatjuk meg. Az 1. opció választása esetén adatainkat a billentyűzetről vihetjük be, míg a 7. opció segítségével egy előzetesen kimentett adatállomány hívható be a kazettás magnetofonról.
A gyári kazettán a "viewpoint" program után három demo található:
"letter" - egy tömzsi "A" betű képe
"cube" - egy kocka képe
"alanine" - egy háló modell az "amino acid L-alanine" képletéről

Ezek szintén a 7. opcióval hívhatók be.

Először is nézzük meg, hogyan történik az adatok manuális bevitele. Tekintsük meg ezt egy kocka felrajzolásán keresztül.



Számokkal nevezünk el minden olyan pontot, ahol két vagy több él találkozik. A koordináta rendszer origója lesz a kiinduló pont (1). A Viewpointban úgy kell gondolkodni, hogy az x tengely balról jobbra, az y tengely alulról felfelé, míg a z tengely tőlünk a papír síkja felé halad, úgy mint a jobb oldali ábrán. Így jelenik meg a testünk a képernyőn is (a bal oldali ábra és koordináta rendszer csak szemléltető jellegű). Aggodalomra viszont semmi ok, a Viewpoint a koordináta tengelyek elforgatása nélkül, vagyis a jobb oldali vázlatnak megfelelően is tudja ábrázolni a 3D képet, s ezt változatos forgatási mechanizmusa révén tudja elérni.
Minden egyes számozott koordináta-ponthoz három érték tartozik,pl.az origóé (0,0,0). Példánkban az egyszerűség kedvéért 1 egységnyi értékeket használunk. A program rugalmasságát bizonyítja az arányok beállítása, és a képernyőterületre történő megfelelő beigazítás, így a testünk a megadott méretek egymás közti arányát biztosítva mindig megfelelő nagysággal jelenik meg a képernyőn.

Az 1. opció választásakor a gép először is meg fogja kérdezni tőlünk a csúcspontok számát (How many points?). 16K-s gép esetén max. 43-at, míg 48K-s gép esetén max. 770 pontot adhatunk meg.

Jelenlegi példánkban a pontok száma 8, így ezt a számot adjuk meg. Az adat begépelése után meg kell nyomni az ENTER-t.

Ezután bekéri a gép az egyes pontok koordinátáit, adatonként mindig nyomjuk meg az ENTER-t.

Végezzük el az adatbevitelt:

	x	y	z
1.pont	0	0	0
2.pont	1	0	0
3.pont	0	1	0
4.pont	1	1	0
5.pont	0	0	1
6.pont	1	0	1
7.pont	0	1	1
8.pont	1	1	1

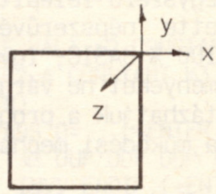
A pontokból még nem tudja eldönteni a gép, hogy a test hogyan épül fel, ezért meg kell adnunk azt is, mely pontok kapcsolódnak egymáshoz egy összekötő egyenes segítségével. A gép egymás után végigkérdezi az összes csúcspontot, de hamar belátható, hogy kétszer nincs értelme ugyanazt a kapcsolatot definiálnunk (pl. az 1.pont kapcsolódik a 2.-ponthoz és a 2. pont is kapcsolódik az 1.-ponthoz), így szimpla ENTER megnyomása ebben az esetben elegendő.
Az egyszerűbb megértéshez nézzük, hogyan járunk el példánkban:

1. pont kapcsolódik	2 E 3 E 5 E
2. pont kapcsolódik	4 E 6 E E
3. pont kapcsolódik	4 E 7 E E
4. pont kapcsolódik	8 E E
5. pont kapcsolódik	6 E 7 E E
6. pont kapcsolódik	8 E E
7. pont kapcsolódik	8 E E
8. pont kapcsolódik	E

Megjegyzés: Az adatbevitelt a "point x connects to" kérdésre kell véghezvinni, ahol

x a kiinduló pont száma.
Példánkban az "E" jelzés az ENTER billentyű megnyomását jelzi.

Az adatbevitel után ismét megjelenik a menü. A program vezérlése ebből a menüből történik. Ha hibaüzenettel leállna a program, vagy BREAK-kel megszakítottuk a program futását, soha sem RUN-nal indítsuk újra, mert a memóriában levő változók törlődnek, s vele együtt a már megkonstruált 3D ábránk is. Ezért a menü lehívásához egyszerűen gépeljünk GO TO 1-et.
Válasszuk most a 2. opciót, a pont körüli forgatást. A gép fel fogja rajzolni az x-y nézeti képet, majd megkérdezi, hogy melyik pont körül forgasson. Példánkban adjuk meg a jobb felső (4.) pontot. Az adott helyen vázlatosan megjelenik a koordináta tengelyek irányjelzése.



A következő kérdésre (Is this the right point?) választásunkat meg is kell erősíteni (Y/N). Nemleges válasz esetén új pont bevitele szükséges. Ha jól választottunk, a gép sorban megkérdezi, hogy az adott pont relatív helyzetéhez képest a testet x,y majd z tengely körül hány fokkal forgassa el? A paraméterek megadása után felrajzolja az elforgatott testet, ezután a "Hard copy?" kérdésre Y/N választással tetszés szerint a képet minden olyan nyomtatóra kimásolhatjuk, amely ismeri a COPY parancsot. Ezt követően meg kell adnunk azt, hogy ezt a képet választjuk-e az új x-y startpozíciónak (Make this the new starting position?). Y/N nemleges választás esetén x-y-ban az eredeti helyzet marad. Végül az "Another option?" lehetőséget ad a menü visszahívására, ha választásunk (Y/N) nemleges, úgy a program futása megáll 0 OK, 930:1 üzenettel.

Ha visszakértük a menüt, válasszuk a 3. opciót, ez két pont által határolt egyenes körül tudja elvégezni a test forgatását. Az első kérdésre a két pont számát kell megadnunk, ezt meg kell erősíteni, majd az "Amount of rotation in degrees" kérdésre fokban be kell állítani a forgatás szögértékét. Ettől kezdve az előző opció szerint jár el a program. Felrajzolja az elforgatott képet, hard copy-t enged meg, igény szerint átállítja az x-y startpozíciót, végül kiléphetünk a programból, vagy visszatérhetünk a főmenübe.
A 4. opció segítségével áthelyezhetjük az x-y startpozíciót tetszés szerinti helyzetbe, megadott két pont által kijelölt irány szerint. A bekért két pontot összekötve megtekinthetjük a két pont kapcsolatát, majd a választás megerősítését követően a megadott irányt a két pont relatív forgatásával állítja át

úgy, hogy az először megadott pontot az előtér felé, míg a másodiknak megadott pontot a hátsó tér felé mozgatja el.

Az 5. opció a mindenkori aktuális x-y nézeti képet (startpozíciót) hívja elő a képernyőre.

A 6. opcióval kimenthetjük a memóriában levő adatainkat. A kimentés előtt a gép bekéri a file nevét, majd a file-t adattömb formájában menti ki.

3D ábránk adattárolásához a gép alapvetően két tömböt használ, a c() és g() tömböket. A c() tömb a startpozíciók adatait tartalmazza, míg a g() tömb a forgatás utáni pozíciók adatait. Mindkét tömb 6-szoros mélységű (x,y,z és 3 kapcsolat). A program használja még a c() tömböt is kétszeres mélységben. A c(1,1) elem tárolja a csúcspontok számát, míg a c(2-6,1) elemek üresek. Az adatállomány kimentésekor a c() tömb is szalagra kerül.

A program főbb részeinek rövid áttekintése:

100-105	-	tömbdefiníciók
106-120	-	koordináta pontok adatbevitele
121-137	-	pontkapcsolatok adatbevitele
150-198	-	menü-kezelés
210	-	adatállomány kimentése
250	-	új adatállomány betöltése
300-320	-	a 2. opció által kijelölt adatok eltárolása
400-420	-	a 3. opció által kijelölt adatok eltárolása
500-520	-	a 4. opció által kijelölt adatok eltárolása
600-650	-	c() és g() tömb összeigazítása
700-811	-	a test felrajzolása a g() tömbben megadott adatoknak megfelelően. Ha pl. molekula-modelleket akarunk forgatni, használhatjuk a CIRCLE utasítást a 711. sorban.
812-813	-	hard-copy a nyomtatóra
820-860	-	ha szükséges, megváltoztatja a startpozíciót
900-930	-	vissza a menühöz, vagy a program vége
1000-1601	-	rajzolás a kiinduló adatok alapján
6000-6005	-	2.opció - kalkuláció végrehajtása
8000-8005	-	3.opció - kalkuláció végrehajtása
9000-9003	-	4.opció - kalkuláció végrehajtása
9110-9200	-	Arány beállítása és a kép beigazítása

Végül egy megjegyzés: a BASIC struktúra egyszerűen megengedi a módosítást, hogy programunk a microdrive-ot is kezelje.

Wonderboy

Az örökélet csak akkor működik, ha a program egy fejléces CODE-ból, egy fejléces SCREEN-ből és egy 33759 byte-os részből áll (plusz még amit külön kell betölteni). Örökélethez a 34362-es címen 0-át kell elhelyezni. Töltsük be a LOADER-t, majd RESET-eljük a gépet, és írjuk be a következő BASIC programot:

```
10 CLEAR 24575: LOAD"" CODE 65400
20 FOR i=65458 TO 65464: READ a: POKE i,a: NEXT i
30 RANDOMIZE USR 65400
40 DATA 175,50,58,134,195,0,128
```

Futtassuk a programot (RUN+ENTER) és indítsuk el a magnót.

Nightmare Rallye

A játék indulása után nyomjuk meg a (SYMBOL SHIFT)-et és a (Q) billentyűt egyszerre. Ekkor az autónk mindenen keresztül mehet és csak kormányozni tudunk. A sebességmutatónk pedig körbe-körbe jár. A következő pálya elején ismét eljátszhatjuk olcsó kis trükkünket!

5. SZEMBEN A SPEEDLOCK-KAL

Folyik a harc a programvédelmi rendszerek, és a "feltörők" között. Az alábbiakban az egyik legelterjedtebb, Spectrumra írt védelmi rendszer, A 'Speedlock' vázlatos ismertetését vettük célba.

A kérdés etikai oldalával (szabad-e, illik-e, helyes-e feltörni a védelmet, és ezzel szabadon másolhatóvá tenni egy programot) nem kívánunk foglalkozni. Miért érdemes mégis tanulmányozni a 'Speedlock'-ot, különösen akkor, ha a Multiface birtokában bárki kényelmesen másolhat védett játékokat. Úgy gondoljuk, egy védelem megfejtése, működésének megértése sokszor izgalmasabb lehet, mint maga a játék. Azok számára, akik nem idegenkednek a gépi kódú programozástól, számos érdekességet, a Z-80 utasításkészletének szokatlan használatát mutatja be a 'Speedlock'.

Ahogyan Ráth György írja a 'Programvédelmi rendszerek IBM PC-re' c. könyvében, a 'Speedlock' védelmi eljárás kifejlesztése hozzávetőleg annyi munkát igényelhetett, mint egy akkori IBM PC-s fejlesztés, a 'Prolok'. A különbség csupán az, hogy az előbbi a 'Match Day' c. játékprogramot védte, melynek ára néhány hét után kb. 5 angol Font volt. Az utóbbi pedig a dBase III. programcsomag védelmére szolgált, melynek ára ma is 250-500 angol Font között mozog.

Magát a védelmi rendszert logikailag két részre bonthatjuk: a betöltő rutin más szinkronjellel, és más baud-sebességgel működik, mint a Spectrum ROM-rutin; Emellett ez a rutin meglehetősen "elrejtett". A betöltő részletes ismertetése nem célunk, arra próbálunk választ találni, hogyan juthatunk el odáig, hogy a betöltőnek ezt az elrejtését fel tudjuk oldani.

A rendszert egy konkrét példán az Ocean cég 'Rambo' c. játékprogramján keresztül ismertetjük (egyszerűen azért, mert ez az egyike azon kevés programnak, amely eredeti változatban áll rendelkezésünkre). Tudomásunk szerint a 'Speedlock'-nak van újabb változata is, célkitűzésünket tekintve ez a lényegen nem változtat.
A 'Rambo' kattogós része előtt két 'BASIC' jellegű programot találunk: egy 205 byteos rövidebb, és egy 1770 byte-os hosszabb részt. Próbáljuk meg először a rövidebbet megfejteni.

A 'MERGE' nem használ; valamilyen másolóprogrammal - vagy egyéb toolkit-tel - ki kell irtanunk az automatikus indítást. A megállított program betöltése után a lista nem sokat mutat a képernyőn. Annál inkább a nyomtatón:

```
0 PAPER 0: INK 0: BORDER 0: CLS:
  PRINT USR 6746931: LOAD""
```

Minden érthetőnek látszik, egy dolog kivételével: nem valószínű, hogy a 'PRINT USR' utáni cím valódi.
Tudjuk, hogy a BASIC rendszer a számokat úgy tárolja, hogy a számok szöveges alakja utáni "0E" karaktert követően található a szám memóriában elhelyezett 5 byteos alakja. Ez természetesen lehet egészen más is, amit a szám szöveges alakja jelent. Bármilyen monitorprogram segítségével azonban megtalálhatjuk a gépi kódú rutin valódi indítási címét. Esetünkben a következőt találjuk:

```
5CFA    8F
5CFB    3A
5CFC    2A
5CFD    33
5CFE    33
```

Ez már kissé zavaró. Ha az indulási cím egész szám volna, az 5 byte-ból csak a harmadik és a negyedik lenne nullától különböző. Milyen címről van tehát szó ? Aki ismeri a Spectrum lebegőpontos számábrázolását, az kézzel is kiszámíthatja. Kényelmesebb azonban a ROM rutinjainak a segítségével meghatározni a címet. Az alábbi rövid program kiírja a megfelelő értéket:

```
LD   HL,#5CFA
CALL #33B4
LD   A,02
CALL #1601
CALL #2DE3
RET
```

Figyeljük meg, hogy nem egész számról van szó. A Spectrum rendszer viszont kerekít, tehát az indulási címet megtaláltuk: 23829. Ha ettől a címtől próbáljuk visszafordítani a programot, a közhasználatú monitorok többsége csődöt mond. Ennek az az oka, hogy a rutin számos "nem standard" Z-80 utasítást tartalmaz. (Ezek részletes ismertetését l. Krizsán György 'Zilog mikroprocesszor családok' c. könyvében - LSI ATSz. 1984.) Olyan utasításokról van szó tehát, amelyek az IX 16 bites regiszter felső és alsó byte-jait külön kezelik. Az idézett könyv jelölése-

it használva a disassembly a következő:

```
5D15   CD 1A 5D    CALL #5D1A
5D18   D9          EXX
5D19   F5          PUSH AF
5D1A   DD 2E 00    LD   XL,0
5D1D   0E FA       LD   C,#FA
5D1F   DD 7C       LD   A,XH
5D21   F3          DI
5D22   06 87       LD   B,#87
5D24   D9          EXX
5D25   DD 2D       DEC  XL
5D27   E6 34       AND  #34
5D29   27          DAA
5D2A   DD 5D       LD   E,XL
5D2C   C1          POP  BC
5D2D   3E F8       LD   A,#F8
5D2F   DD 65       LD   XH,XL
5D31   16 5C       LD   D,#5C
5D33   D9          EXX
5D34   51          LD   D,C
5D35   DD 6F       LD   XL,A
5D37   3E 1E       LD   A,#1E
5D39   DD F9       LD   SP,IX
5D3B   80          ADD  A,B
5D3C   FD CB 01 A6 RES  4,(IY+01)
5D40   2E 24       LD   L,#24
5D42   D9          EXX
5D43   DD 6B       LD   XL,E
5D45   3C          INC  A
5D46   1E 3E       LD   E,#3E
5D48   ED 4F       LD   R,A
5D4A   D9          EXX
5D4B   EE 99       XOR  # 99
5D4D   DD 62       LD   XH,D
5D4F   DD 55       LD   D,XL
5D51   1E 3D       LD   E,#3D
5D53   ED 47       LD   I,A
5D55   DD 7C       LD   A,XH
5D57   FD 77 03    LD   (IY+03),A
5D59   03          INC  BC
5D5A   DD 2E 98    LD   XL,#98
5D5D   7A          LD   A,D
5D5E   D9          EXX
5D5F   12          LD   (DE),A
5D60   3E FF       LD   A,#FF
5D62   1E 69       LD   E,#69
5D64   DD 67       LD   XH,A
5D66   0A          LD   A,(BC)
5D67   DD 77 00    LD   (IX+00),A
5D6A   03          INC  BC
5D6B   DD 23       INC  IX
5D6D   1D          DEC  E
5D6E   C2 66 5D    JP   NZ,#5D66
5D71   D9          EXX
5D72   FB          EI
5D73   C9          RET
5D74   10 1C       DEFB #10,#1C
5D76   52 1B       DEFB #52,#1B
5D78   76 1B       DEFB #76,#1B
5D7A   03 13       DEFB #03,#13
5D7C   00 3E       DEFB #00,#3E
```

A rutin számos -látszólag felesleges - utasítást tartalmaz. Ha a második résszel kezdjük a betöltést, a játék nem működik. Melyek tehát a lényeges részek? Az #5D39-es címen található LD SP,IX utasítás, amely a veremmutatót #FFF8-ra állítja, továbbá az #5D66-#5D70 tartományon lévő ciklus, amely a verem területét felülírja; emiatt a rutin végül az STMT-RET (#1B76) ROM-rutinon keresztül tér vissza a BASIC rendszerhez. Ez a program tehát kiváltható egy CLEAR 65533 paranccsal (ami persze a RAMTOP-ot is átállítja, a program nem). A 'Rambo' valóban működik, ha a fenti CLEAR után a betöltést a második, 'bl' nevű file-jával kezdjük: ebben újabb 'trükkökkel' fogunk találkozni.



Töltsük be a játékot, és vegyük fel magnóra a játék állását, majd RESET és írjuk be a következő BASIC-et:

```
 1 REM"15 db. 0"
10 CLEAR 24999
20 FOR i=23760 TO 23774: READ a: POKE i,a: NEXT i
30 DATA 221,33,96,234,17,102,0,62,255,55,205,86,5,251,201
40 RANDOMIZE USR 23760: POKE 60018,16: POKE 60019,39: POKE 60020,0:
   POKE 60021,0
50 SAVE"*" CODE 60000,102
```

Indítsuk el a programot (RUN+ENTER), és töltsük be az előbb felvett játékállást. Nyomjunk meg egy billentyűt, ekkor ez aktivizálódik. Visszamentéskor csak a fejléc nélküli részt vegyük fel a magnóra. Töltsük be újból a játékot, és töltsük vissza imént felvett byte-jainkat. A játékban 'TÍZMILLIÓ' egység pénzünk lesz. Ezt a kis átalakítást többször is megtehetjük.

A RAM Disc

Kicsit meredeknek tűnik az a kijelentés, hogy a 128K Spectrumon úgy használhatjuk ki a +80K memóriát (RAM disc), mint egy előre formattált lemezt, vagy microdrive kazettát.

Mint tudjuk a Z-80 mikroprocesszor egyidőben mindössze csak 64 kbyte memóriaterületet tud kezelni, a 128K Spectrum további memóriáinak eléréséhez két lehetőségünk kínálkozik.

Az első, a teljes memória egyidőben történő kihasználása, amelyet BASIC-ből nem tudunk megoldani, ez csak gépi kód használata esetén hasznos, a lapozó rutinok felhasználásával (erről később lesz szó).

A második és könnyebb lehetőségünk a háttér memória - mint tároló (a továbbiakban RAM disc) - BASIC-ből történő elérése.

A RAM disc kezeléséhez a 128K BASIC üzemmódban több új BASIC utasítást vezettek be, a továbbiakban először ezeket tekintjük át:

SAVE ! "abc"
: Elmenti a RAM disc-re az 'abc' nevű BASIC programot.

SAVE ! "abc" LINE xx
: Elmenti a RAM disc-re az 'abc' nevű BASIC programot az xx. sortól automatikus indítással.

SAVE ! "abc" CODE xx,yy
: Elmenti a RAM disc-re az 'abc' nevű gépi kódú programot (memóriatartalmat) az xx. címtől kezdve, yy. byte hosszon.

SAVE ! "abc" SCREEN$
: Elmenti a RAM disc-re az 'abc' nevű képernyő-memória tartalmat.

SAVE ! "abc" DATA n()
: Elmenti a RAM disc-re 'abc' néven az n() adattömböt.

SAVE ! "abc" DATA n$()
: Elmenti a RAM disc-re 'abc' néven az n$() string-tömböt.

LOAD ! "abc"
: Betölti a RAM disc-ről az 'abc' nevű BASIC programot.

LOAD ! "abc" CODE (xx,yy)
: Betölti a RAM disc-ről az 'abc' nevű gépi kódú programot (memóriatartalmat), xx. és yy. memória-kezdőcím és hossz paraméterek megadhatók.

LOAD ! "abc" DATA n()
: Betölti a RAM disc-ről 'abc' néven az n() adattömböt.

LOAD ! "abc" DATA n$()
: Betölti a RAM disc-ről 'abc' néven az n$() string-tömböt.

MERGE ! "abc"
: Összefésüli a memóriában található és a RAM disc-ről 'abc' névvel betöltött BASIC programot. Ha azonos sorszámok fordulnak elő, a memóriában lévők törlődnek.

CAT !
: Kilistázza a képernyőre a RAM disc katalógusát.

ERASE ! "abc"
: Törli a RAM disc-ről az 'abc' nevű file-t.

Hiba a 128K BASIC-ben!

A 128K BASIC Interpreter, mint ismeretes, nem kulcsszavakkal dolgozik, hanem karakterenként kell bebillentyűznünk az egyes utasításokat. Matematikai kalkulációk esetén - a karakterek tokenizálásakor - egy bizonyos esetben hibajelenséggel találkozhatunk.

Gépeljük be:

```
10 IF A>B-C THEN STOP
```

ENTER után a képernyőn a következő jelenik meg:

```
10 IF AB> -C THEN STOP
```

Nem kell megijedni, mert ettől még a programunk hibátlanul fog lefutni. Listázáskor viszont ez a forma zavaró lehet, ezért célszerű a sort átírni:

```
10 IF B-C<A THEN STOP
```

vagy így is eljárhatunk:

```
10 IF A>(B-C) THEN STOP
```

Megjegyzés: A probléma a 48K üzemmódban nem jelentkezik !

Zub

A program file-térképe a következő: 128/6912/40836.
Ha azt akarjuk, hogy ne legyen ellenség, töltsük be a LOADER-t, majd RESET-eljük a gépet, és írjuk be a következő BASIC programot:

```
10 CLEAR 24699: LOAD"" CODE
20 POKE 37473,201: RANDOMIZE USR 56576+USR 24700
```

Futtassuk a programot (RUN+ENTER) és indítsuk el a magnót.

A perifériák királya ? - tették fel sokan a kérdést néhány évvel ezelőtt, amikor a ROMANTIC ROBOT UK.Ltd. megjelent a piacon "varázsdobozával".

Mit is tud valójában ez a készülék ? Nos elsődleges és legfontosabb tulajdonsága a teljesen univerzális és 100 százalékosan automatikus SAVE funkció, amelyet nem csak magnetofonra tud végrehajtani, hanem microdrive-ra és a legfontosabb lemezes perifériákat kezelő Interface-ekre is. Ezen túlmenően tervezői ellátták KEMPSTON típusú joystick csatlakozóval, valamint helyet kapott a készülékben 8K szabadon kihasználható RAM memória is, amely teljesen hozzáférhető, és a 128K Spectrum RAM disc-hez hasonlóan kezelhető.
Ha rendelkezünk ilyen készülékkel nem árt, ha megfogadunk néhány tanácsot, ugyanis a készüléken elhelyezett áramköri elemek hamar meghibásodhatnak, s az esetleges alkatrészcsere gondokat okozhat.
Először is, mielőtt a MULTIFACE-t a Spectrumra illesztjük, áramtalanítsuk a számítógépet, mivel ellenkező esetben készülékünk tönkre mehet. Ha a készülék illesztése megtörtént, a MULTIFACE kapcsolóját váltsuk 'ON' állásba, máskülönben működésképtelen. Ha joysticket is akarunk használni, úgy annak dugóját is csatlakoztassuk a MULTIFACE megfelelő aljzatába. Ha minden kész, bekapcsolhatjuk a számítógépet. Sajnos a MULTIFACE több verziója látott napvilágot, sőt igen sok "maszek" tákolmány is kifutott a felhasználók közé, ami nem mindig garantálja a hibátlan inicializálást, ezért előfordulhat, hogy a rendszer nem jelentkezik be. Próbáljuk megnyomni a MULTIFACE varázsgombját. Amennyiben véletlen színkombinációkon kívül más nem jelenik meg a képernyőn, áramtalanítsunk, húzzuk le a MULTIFACE-t az él-csatlakozóról, majd a bekapcsolási műveletet kezdjük előről. Ha a rendszer ismételten sem állna fel, még próbálkozhatunk egy párszor, de előbb utóbb rá kell jönnünk, hogy a MULTIFACE javításra szorul, keressük meg a legközelebbi szakembert problémánkkal.

A MULTIFACE funkciói

A MULTIFACE aktivizálásához először is meg kell nyomnunk a varázsgombot, aminek hatására a legalsó sorban kis menü jelenik meg a képernyőn.
A funkció kiválasztásához meg kell nyomnunk a megfelelő billentyűt.
A következőkben tekintsük át a lehetséges funkciókat:

(e)xit - 'meleg RESET' + kilépés a BASIC rendszerhez

Segítségével kiléphetünk a BASIC rendszerhez, de úgy, hogy a programunk épségben a memóriában marad. Ennek a dolognak az egyetlen alapfeltétele, hogy a normál Spectrum rendszerváltozók a memóriában sértetlenül megmaradjanak, máskülönben a rendszer sajnos "lefagyhat".
Sikeres kilépés esetén teljes hozzáférhetőségünk lesz a programhoz, abban változtatásokat eszközölhetünk, majd a MULTIFACE varázsgomb megnyomásával visszatérve a MULTIFACE főmenühöz, a módosított programot kimenthetjük.
Ha a programot akarjuk újraindítani, akkor ismernünk kell a start sort, vagy címet.

(r)eturn - a program folytatása

Ha egy futó (játék)programot megállítunk a MULTIFACE varázsgomb megnyomásával, és dolgainkat elvégeztük, a program futását a (r)eturn funkció kiváltásával engedhetjük tovább.

(s)ave - mentő rutinok aktivizálása

Először is meg kell adnunk a program nevét. Csak 9 karakter van engedélyezve (BETA Interface felhasználása esetén pedig csak 7), majd meg kell nyomni az ENTER-t.
Újabb választás előtt állunk:

(t)ape (c)artridge (w)affer (d)isk

A disk kiválasztásán túl több dolgunk nem lesz, mivel a MULTIFACE magától megállapítja, hogy pl. BETA, vagy DISCOVERY van csatlakoztatva.
Mindezen túl meg kell adnunk, hogy

programot - (p)rogram,

vagy

képernyőt - (s)creen

kell kimenteni.

A kimentéskor a MULTIFACE tömörít, vagyis a háttértároló eszközön a futó programnak csak egy tömörített változata lesz elhelyezve, ebből adódóan a betöltési idő is lecsökken (főként a kazettás magnetofon esetében). Mindezt figyelembe véve megállapíthatjuk, hogy a tömörített kód egyszerű disassemblálása nem lehetséges.

(t)ool - MULTIFACE TOOLKIT rutinok meghívása

(q)uit -visszatér a MULTIFACE főmenübe

ENTER+SPACE - PEEK/POKE funkció

SPACE törli az input mezőt, ekkor egy memóriarekesz címének megadását várja a gép. Ha begépeltük a címet, azonnal megjelenik a rekesz aktuális tartalma. ENTER megnyomására tovább lépünk a következő memóriacímre, egyébként átírhatjuk a tartalmat, amelyet szintén ENTER-rel érvényesíthetünk.

(h)ex - átváltás a hexadecimális és decimális kijelzés között

(r)eg - kijelzi a processzor regisztereinek aktuális értékeit.

A regisztertartalmakat a decimális 16358-16383 címekre másolja át.

(w)indow - MONITOR ablak megnyitása

A képernyő felső részén megnyit egy 128 byte méretű ablakot, melyben az aktuális memóriacímet követő 128 rekesz tartalmát tekinthetjük át. A megfelelő, és átírandó cím a kurzorbillentyűk használatával választható ki.

(t)ext - ASCII ablak megnyitása

Az előző funkciótól annyiban különbözik, hogy a memóriarekeszek tartalma ASCII karakterek formájában jelenik meg a képernyőn.

(c)opy - Képernyő-tartalom kinyomtatása

Kinyomtatja az aktuális képernyő-tartalmat olyan nyomtatóra, amely ismeri a COPY parancsot (pl. LPRINT III., KEMPSTON 'E', stb.).

(j)ump - Ugrás egy új címre

Nem egy új címen kezdődő programfuttatást valósít meg, hanem egy új címmezőre való átugrást, további manipulációk céljából. Az ugrási cím a decimális 8192/8193 címeken tárolódik el. Az ugrási cím a teljes Spectrum ROM/RAM tartományba mutathat, továbbá a MULTIFACE 8K RAM-ba is.
A MULTIFACE 8K RAM átfedi a Spectrum ROM-ot (8192-16383), a lapozási státuszt a 8194. címen találjuk. Ha ezen a címen zérus van, az alap ROM működik, 1 esetén pedig a MULTIFACE RAM.

Army Moves

Töltsük be a játékot. Nyomjuk meg a 4-es billentyűt és az "ACCESS CODE?" kérdésre üssük be a 27351 számot, és indítsuk el a magnót. Így a program második részével is játszhatunk anélkül, hogy az elsőt végigjátszottuk volna.

Xevious

A program felépítése: BASIC, és egy fejléc nélküli, 48886 byte-os rész. 255 db. élethez töltsük be a LOADER-t, majd BREAK-eljük le. Írjuk be a következő parancsokat: POKE 65417,62: POKE 65418,255: POKE 65419,50: POKE 65420,88: POKE 65421,208 (ENTER)
RANDOMIZE USR 65400 (ENTER), és indítsuk el a magnót.

Eltűnt bit(ek)?

Amennyiben rossz minőségű magnetofonnal dolgozunk, gyakran előfordulhat, hogy kölcsönadott kazettáinkról barátaink igen kevés programot tudnak átmásolni. Ugyanez a gond velünk is előfordulhat, kölcsönkapott kazetta esetén.
A beolvasási problémák legismertebb tünete a 'Tape loading error' hibaüzenet megjelenése. Ennek az üzenetnek igen sok forrása lehet, mi most itt csak egy konkrét hibaforrás feltárását és a hiba kijavításának egy lehetséges módszerét ismertetjük.
Ez a hiba az ún. 'bithiba', amely gyakran előfordulhat, főként akkor, ha automatikus felvételi szintszabályzós magnetofonnal történik a programok rögzítése. A 'bithiba' azt jelenti, hogy a szalagon tárolt adatmező elejéről hiányzik egy vagy több bit, ami felborítja a teljes program-struktúrát (a byte-ok felépítése 8 bitenként történik). Ez a hiba maga után vonja az illető adatblokk ún. 'markbyte'-jának megsérülését is, ez pedig azt fogja okozni, hogy a ROM-beli LOAD rutin a betöltést hibaüzenettel szakítja meg.
Első dolgunk megvizsgálni, érdemes-e a programot 'megmentenünk'. Értékesebb program esetén fontos, hogy legalább a BASIC loader ép legyen, mert a további információkat főként innen fogjuk tudni kiolvasni. Fontos természetesen azt is figyelembe venni, hogy az illető program ismételt beszerzése gondot okoz-e ? Ha igen, próbáljuk meg a 'műtétet'.

Először is célszerű lépésről-lépésre levizsgálni a program egyes file-jait (több file-ból álló program esetén). Ezt másolóprogram segítségével a legegyszerűbb elvégezni. Meg kell állapítanunk, hogy az említett hiba csak egy file, vagy minden további file esetén fennáll-e. Ez utóbbi csak a 'műtét' idejét fogja megnövelni, ill. csökken a 'műtét' sikerének az esélye.

A konkrét 'műtétet' tekintsük meg egy vázlatos példán keresztül:
Kölcsönkazettán 2 részből álló program vár megmentésre. Mindkét file fejléces, az első a BASIC loader, míg a második a kód. Betöltés közben - a kód fejlécét követő TONE után - a betöltés hibaüzenettel áll le.
Első feladatunk, hogy betöltünk egy másolót - pl. SUPER 50K -, és megvizsgáljuk, hogy a fejlécben megadott byte-ok száma megegyezik-e a kód tényleges hosszával. A bithiba legismertebb tünete, hogy a végérték eggyel kevesebb, mint amennyit a fejléc tartalmaz. (Fejléc nélküli kód esetén célszerű a loader-ből kiolvasni a tényleges értéket, és ezzel összevetni a valódit.)
Példánkban a kód kezdődjön a 25700. címen, és hossza legyen 30000. byte. A másoló csak 29999 byte-ot jelez ki, tehát 'nyomon vagyunk'. Ezután meg kell határoznunk a 'markbyte' értékét. A 'markbyte' normálistól való eltérésére az is felhívhatja a figyelmet, ha a másolóba betöltött 'bithibás' kódot megpróbáljuk kimenteni, és a fejléc utáni TONE a szokásostól eltérő hosszúságú. A SUPER 50K nem jelzi ki nekünk a 'markbyte' értékét, de pl. az OMNICOPY-t betöltve azonnal kijelezhetjük ezt az értéket. Példánkban az OMNICOPY 99-et jelez, nem is csoda, hogy a ROM loader 'kiakadt', hiszen 255-öt várt.
A 'markbyte' ismeretében kíséreljük meg a 'bithibás' kód betöltését - most fejléc nélkül, és hibás 'markbyte'-tal -. Ehhez gépeljük be és futtassuk a következő BASIC programot:

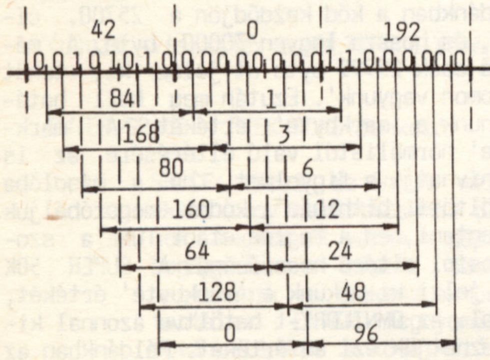
```
10 CLEAR 65299
20 FOR i=65300 TO 65313
30 READ a: POKEi,a: NEXT i
40 DATA 221,33,L1,H1,17,L2,H2,
   62,M,55,205,86,5,201
```

Programunkban L1 és H1 a kód töltési kezdőcímének alsó és felső byte-ja (esetünkben L1=100; H1=100), L2 és H2 a kód hosszának alsó és felső byte-ja (esetünkben L2=48; H2=117), végül M a 'markbyte', ami esetünkben éppen 99.

Ha a paramétereket beírtuk, futtassuk a BASIC-et (RUN+ENTER). Ez elhelyezi a 65300. címen kezdődő betöltőt, s ha kiadjuk RANDOMIZE USR 65300 (ENTER), indíthatjuk a magnetofont a kód fejlécet követő TONE-jától. A 'bithibás' kód betöltődik a helyére, ám erről a 'bithibáról' meg is kellene győződnünk. Értelmes gépi kódot ott érdemes keresni, ahol a rutinnak a belépési pontja van, mert elképzel-

hető az is, hogy a kód elején éppen képernyőt, vagy tetszőleges adatmezőt helyeztek el. Ez a belépési cím a legtöbb esetben ugyancsak a loader-ből olvasható ki. Ha a kódunk mérete megengedi, még egy disassemblert is betölthetünk mellé, így könnyebb lesz a dolgunk, ha ez nem megoldható, kénytelenek vagyunk az első néhány byte alapján mi visszafejteni a kódot.

A bitek elcsúszásából eredő effektust tekintsük meg egy ábra segítségével:



Az eredeti – egymást követő három byte – az LD HL,(49152) utasítás három kódja. Gépi kódú rutinok belépési pontjainál igen sok esetben találkozunk töltő utasításokkal. Ha egy vagy két bit lecsípődik az elejéből, láthatjuk milyen 'galibát' okozhat.

Attól függően, hogy hány bit 'csípődött' le a kód elejéből, rendbe kell szednünk a byte-okat, erre a legcélszerűbb egy bitléptető mechanizmust alkalmaznunk:

```
 10 DIM c(10)
 20 LET z=0: LET c=0
 30 FOR i=a TO b
 40 LET a=PEEK i
 50 GO SUB 100
 60 POKE i,z
 70 NEXT i
 80 STOP
100 LET b=a/128
110 IF b=1 THEN LET c(7)=1: GO TO 330
120 IF b>1 THEN LET c(7)=1: LET a=a-128
130 LET b=a/64
140 IF b=1 THEN LET c(6)=1: GO TO 330
150 IF b>1 THEN LET c(6)=1: LET a=a-64
160 LET b=a/32
170 IF b=1 THEN LET c(5)=1: GO TO 330
180 IF b>1 THEN LET c(5)=1: LET a=a-32
190 LET b=a/16
200 IF b=1 THEN LET c(4)=1: GO TO 330
210 IF b>1 THEN LET c(4)=1: LET a=a-16
220 LET b=a/8
230 IF b=1 THEN LET c(3)=1: GO TO 330
240 IF b>1 THEN LET c(3)=1: LET a=a-8
250 LET b=a/4
260 IF b=1 THEN LET c(2)=1: GO TO 330
270 IF b>1 THEN LET c(2)=1: LET a=a-4
280 LET b=a/2
290 IF b=1 THEN LET c(1)=1: GO TO 330
300 IF b>1 THEN LET c(1)=1: LET a=a-2
310 LET b=a/1
320 IF b=1 THEN LET c=1
330 IF c=1 THEN LET z=z+128
340 IF c(7)=1 THEN LET z=z+64
350 IF c(6)=1 THEN LET z=z+32
360 IF c(5)=1 THEN LET z=z+16
370 IF c(4)=1 THEN LET z=z+8
380 IF c(3)=1 THEN LET z=z+4
390 IF c(2)=1 THEN LET z=z+2
400 IF c(1)=1 THEN LET z=z+1
410 RETURN
```

A mechanizmust az egyszerűség kedvéért BASIC-ben írtuk meg. A 30. sorban kell 'a' és 'b' paraméterek helyén beállítanunk a kód kezdő és végcímének értékét (példánkban a=25700 és b=55699). A bitcsúsztatást a 100. sortól elhelyezkedő szubrutin végzi el. Az új kód az előző helyén fog kialakulni. Ha így sincs értelmes kód, a bitcsúsztatást még hatszor elvégezhetjük. Ha ezután sem járnánk sikerrel, akkor a probléma nem ilyen egyszerű (most itt nem célunk a további lehetőségeket feltárni).

A bitcsúsztatás végrehajtása ezzel a BASIC segédprogrammal meglehetősen lassú, aki ilyen problémával találkozik, megpróbálhatja a léptető rutint átírni gépi kódba, ott megkönnyíti a dolgunkat az, hogy a Z-80 ismer bitléptető utasításokat.

Ha helyreállítottuk a kódot, azt ezek után normál BASIC SAVE segítségével kimenthetjük (természetesen, ha eredetileg a kód fejléc nélküli volt, csak a fejléc utáni részt mentjük ki). Az összeállított file-okat ellenőrizzük le, majd töltsük be a programot. Az említett hibajelenség esetében ezzel a módszerrel az esetek 80-90 százalékában sikerrel járunk.

A 'Tape loading error' hibaüzenet megjelenésének többszáz oka lehet. Mi most csak egyet emeltünk ki ezek közül, reméljük még így is sok felhasználó tudatosan áll neki a 'bit-mentésnek', ha szükség van rá.

5. A memória adott rekeszének töltése regiszterekből vagy konstanssal

Ha alaposan áttekintettük azt, hogy hogyan történik a regiszterek töltése a memória meghatározott rekeszéből, nem fog problémát jelenteni ezeknek az utasításoknak a könnyebb elsajátítása.
Itt is abszolút, indirekt és indexelt címzéssel találkozhatunk, de ezekben az utasításcsoportokban lehetőségünk van a memóriarekeszek konstans adatbyte-okkal való feltöltésére is.

- Memóriarekeszek töltése abszolút címzéssel

Az előző csoporthoz hasonlóan itt is 'nn', 'NN', 'x', 'y' és 'X', 'Y' jelöléseket használunk a memóriacím ill. alsó/felső byte-jának decimális és hexadecimális azonosításához.

```
       50,x,y      32 X Y     LD  (nn),A    ;az adott memóriahelyre
       34,x,y      22 X Y     LD  (nn),HL   ;beírja a regiszterben, vagy
   237,99,x,y   ED 63 X Y     LD  (nn),HL   ;regiszterpárban található
   237,67,x,y   ED 43 X Y     LD  (nn),BC   ;adatot
   237,83,x,y   ED 53 X Y     LD  (nn),DE
   221,34,x,y   DD 22 X Y     LD  (nn),IX
   253,34,x,y   FD 22 X Y     LD  (nn),IY
  237,115,x,y   ED 73 X Y     LD  (nn),SP
```

Ha jól megvizsgáljuk az utasításokat, hamar kiderül, hogy ebben a csoportban nincs lehetőségünk a konstans adatbyte-ok kezelésére, vagyis nem tehetjük meg azt, hogy pl. egy közvetlen memóriacímet egyetlen utasítás segítségével direkt konstans adatbyte-tal töltsünk fel.
Ha ilyen műveletet akarunk végrehajtani, akkor először is az adatot be kell töltenünk egy regiszterbe, majd az adott regiszter tartalmát már áttölthetjük az illető memóriarekeszbe.
Gyakorlásképpen tekintsünk meg egy egyszerű példát:
Kis vonalszakaszt akarunk megjeleníteni a képernyő bal-felső sarkában. A képernyőmemória aktuális címe 16384 (4000h), az adatbyte értéke pedig legyen 255 (FFh).
Gépi kódban ezt a következőképpen tudjuk elérni:

```
    62,255      3E FF      LD   A,255        ;az 'A' regisztert feltöltjük
                                             ;az adatbyte-tal
    50,0,64     32 00 40   LD   (16384),A    ;az 'A' regiszter tartalmát
                                             ;betöltjük a 16384 (4000h) címre
    201         C9         RET  -            ;rutin vége
```

Ezt a rutint a memória bármely szabad területén futtathatjuk, a továbbiakban a kódokat (gépi kódot) el kell helyeznünk az általunk kiválasztott memóriarekeszekben.
Legyen a rutin számára általunk kiválasztott báziscím 30000 (7530h), így a próbálkozás azok számára sem fog gondot jelenteni, akik csak 16K-s géppel rendelkeznek.
A gépi kód bevitelére számos módszer ismeretes, s tekintettel arra, hogy "a gyakorlás az élet tanítómestere", röviden tekintsük át a bevitel lehetséges módjait:

a/ BASIC-ből

- direkt POKE-olással
  példánkban:

```
POKE 30000,62: POKE 30001,255: POKE 30002,50
POKE 30003,0: POKE 30004,64: POKE 30005,201
```

  Ez a bevitel hosszabb kód esetén megnöveli a hibázás lehetőségét, nem beszélve arról, hogy meglehetősen lassú.

- DATA sor(ok)ban, ciklusból beolvasva
  példánkban:

```
10 FOR i=30000 TO 30005
20 READ a: POKE i,a: NEXT i
```

```
30 DATA 62,255,50,0,64,201
RUN (ENTER)
```

Itt is előfordulhat, hogy hibázunk:

- nem jól adtuk meg a ciklusváltozó végértékét, és kevesebb adatot olvastunk be, ami megnöveli annak az esélyét, hogy lemarad a RET, ezért a gépi kódú rutinból nem tudunk visszatérni a BASIC rendszerhez, következésképpen "lefagy" a gép.
- kevesebb adatot adtunk meg a DATA sorban, ilyenkor a gép 'E Out of DATA' hibaüzenettel leáll.
- elgépeltük az adatot, vagy több adatot adtunk meg. Ez utóbbi hibát a legjobban un. kontroll-összeggel tudjuk kivédeni, vagyis beolvasáskor folyamatosan összegezzük a számokat, és ha a beolvasás végén nem stimmel a végösszeg, a programot mi irányítjuk hibajelzésre:

```
40 IF i<>632 THEN PRINT "Hibas adatbevitel":STOP
```

b/ <u>MONITOR-ból</u>

A monitorok többsége azonos mechanizmus szerint engedi meg a kódok bevitelét. Lehetőségünk van egy adott memóriaterület kilistázására (DUMP). A listázó parancsok különbözőek lehetnek. Egy képernyő méretű memóriatartalom kijelzését követően beléphetünk a szerkesztő (EDITOR) módba, vagyis felülírhatjuk a memória aktuális területét. Ezt általában a hexadecimális számok felhasználásával végezhetjük el. Válasszuk ki itt is a 30000. címet (7530h), majd ettől a címtől kezdve lépjünk be a szerkesztő módba és gépeljük be sorban a megfelelő értékeket:3E FF 32 00 40 C9. Az adatbevitel monitoronként különböző, egyes monitorok az adat begépelését követően a soron következő címre lépnek, más monitoroknál ENTER-t is kell nyomnunk. Minden monitor külön parancs segítségével (pl. Q=quit, X=exit, stb.) lehetővé teszi a BASIC rendszerhez való visszatérést.

c/ <u>ASSEMBLER-rel</u>

Az assembler programok - mint ahogy már ezt megemlítettük - <u>a gépi kódú programok memóriába bevitelének legbiztosabb forrásai</u>. Itt ugyanis nem a konkrét adatbyteokat visszük be közvetlenül a memóriába, hanem a gépi kód forrásszövegét, az un. mnemonikokat írjuk be az assembler programba. A forrásszöveg (SOURCE CODE) önmagában még nem futtatható.

Az assembler programok között is nagy a választék, a bevitelre csak mintákat tudunk ajánlani (később - természetesen az összes utasítás megismerése után - konkrét rendszerprogramokkal is fogunk foglalkozni).

```
10 ORG 30000            0001        ORG 7530
20 LD A,255             0002 START  LD A,FF
30 LD (16384),A         0003        LD (4000),A
40 RET                  0004 END    RET
```

Akár decimális, akár hexadecimális bevitelt vár el az assembler, a forrásszöveg elején minden esetben meg kell adnunk azt a báziscímet, ahol az assembler a forrásszöveg alapján előállítja nekünk a tárgykódot (OBJECT CODE). Ezt a tárgykódot minden assembler egy speciális fordítási parancs segítségével generálja a forrásszövegből. A tárgykód már futtatható. <u>A forrásszöveg alkalmazása megkönnyíti az esetleges hibák kijavítását</u> (DEBUGGING), amit a DATA sorok alkalmazásával sokkal nehezebben tehetünk meg.

A memóriában elhelyezett tárgykód - függetlenül attól, hogy BASIC-ből, MONITOR-ból, vagy ASSEMBLER segítségével került a memóriába - vizsgálatára a legalkalmasabbak az un. DISASSEMBLER programok. Ezek a tárgykódot visszafordítják mnemonikokká, vagyis a gépi kódú programok assembly listáját adják eredményül.

Ha kiválasztottuk a megfelelő beviteli módot, és a kódot is elhelyeztük a memóriában, futtassuk "látványos" rutinunkat:

```
RANDOMIZE USR 30000 (ENTER),
```

melynek eredményeképpen megjelenik a képernyő bal-felső sarkában egy kis vonal-szakasz.
Bizonyos esetekben BASIC-ből egyszerűbben kiválthatunk memóriaműveleteket, pl. a POKE utasítással közvetlenül feltölthetünk memóriarekeszt adatbyte-tal:

```
POKE 16384,255 (ENTER)
```

Megjegyzés: A POKE utasítás hatására a BASIC Interpreter az imént megismert gépi kódú rutint hajtja végre.

- <u>Memóriarekeszek töltése indirekt címzéssel</u>

Ez az utasításcsoport gyakorlatilag ugyanazt a célt szolgálja, mint az előző, csak itt a kiválasztott memóriacím értékét be kell töltenünk egy aktuális regiszterpárba.

```
 119    77     LD  (HL),A   ;az adott regiszterpárba
   2    02     LD  (BC),A   ;beírja a regiszterben
  18    12     LD  (DE),A   ;található, vagy konstans
 116    74     LD  (HL),H   ;adatbyte-ot
 117    75     LD  (HL),L
 112    70     LD  (HL),B
 113    71     LD  (HL),C
 114    72     LD  (HL),D
 115    73     LD  (HL),E
54,n    36 N   LD  (HL),N
```

Előző mintapéldánkat módosítsuk ennek megfelelően:

```
33,48,117  21 30 75   LD  HL,30000
   62,255     3E FF   LD  A,255
      119        77   LD  (HL),A
      201        C9   RET
```

A végeredmény természetesen ugyanaz lesz.
Ennek a változtatásnak az előnyét itt nem érezzük, de majd a matematikai, ill. ciklus utasítások megismerése után konkrét mintapéldával visszatérünk ehhez a csoporthoz.

Ezt a módosítást egyébként BASIC-ben így írhattuk volna le:

```
10 LET HL = 30000
20 LET A = 255
30 POKE HL, A
```

- <u>Memóriarekesz töltése indexelt címzéssel</u>

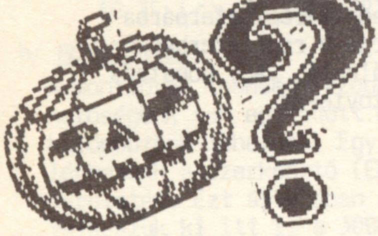
Különösebb kommentár nélkül íme az idetartozó utasítások listája:

```
 221,109,n    DD 77 N     LD  (IX+N),A   ;Az IY utasítások szerkezete
 221,106,n    DD 74 N     LD  (IX+N),H   ;hasonló, csak 221 helyett
 221,107,n    DD 75 N     LD  (IX+N),L   ;253-at, DD helyett FD-t, ill.
 221,102,n    DD 70 N     LD  (IX+N),B   ;IX helyett IY-t kell írnunk.
 221,103,n    DD 71 N     LD  (IX+N),C
 221,104,n    DD 72 N     LD  (IX+N),D
 221,105,n    DD 73 N     LD  (IX+N),E
221,54,n,m    DD 36 N M   LD  (IX+N),M
```

Megjegyzés: Ez utóbbi esetben m (ill.M) a konstans adatbyte értéke.

TARTALOMJEGYZÉK:

THE DAMBUSTERS



1. sz. műszaki fülke:
A jobboldali karok nem a légcsavar áttételét, hanem a lapátok állásszögét változtatják. Teljesen előretolt helyzetben a légcsavarlapátok "kisszögön" vannak, a fordulatszám a gázkarral változtatható. Ha a karokat hátrébb húzzuk, a lapátoknak nagyobb állásszöget adunk, akkor a fordulatszám csökken, mert a motort jobban leterheltük. Az így beállított fordulatszám gázadásra sem változik, mert egy centrifugál regulátor a lapátoknak nagyobb állásszöget ad, tartva a beállított fordulatot. Ezt a légcsavartípust nálunk változtatható állászögűnek hívják, de az angol elnevezés találóbb: constant speed airscrew.

2. sz. műszaki fülke:
A középső kar nem a csűrőlapokat mozgatja, ezek ugyanis a gép bedöntésére (forduló) szolgálnak, és a botkormánnyal mogatjuk azokat. A középső kar a féklapokat, ill. fékszárnyakat mozgatja, kitérített helyzetben nagyobb íveltséget adva a szárnyprofilnak, így a gép kisebb sebességnél sem esik át. (Fel és leszállás.)

Ami tényleg lényeges:
A baloldali kar nem az üzemanyag adagolását végzi, hanem ez az oldalkormánytrimm. (Trimmlap=kormányerő kiegyenlítőlap.) Ezt akkor venni észre, ha egy vagy több motor le van állítva és emiatt a gép vízszintes szárnyakkal is állandóan eltér az iránytól. Ilyenkor ezzel a karral kell a gépet "kitrimmelni", hogy tartsa az iránytűn beállítottt irányszöget.

Végül ami még nem derült ki: A gátakat érdemes a tavaknál keresni.

A szerzők a következő címen érhetők el :
SPECTRUM VILÁG - BUDAPEST-3, *Postán maradó*, 1300

Akiknek nem jutott volna a 'SpV' I. II. ill. III. részéből, azoknak azt ajánljuk, rendeljék meg a fent olvasható címen!

Felelős kiadó: Rucz Lajos, Székely László · K. F. eng.-sz.: 59406
10000 – 807214 Pátria Nyomda
Felelős vezető: Vass Sándor vezérigazgató

39,– Ft